SIMTRUM

电动位移台驱控器使用手册

2023 V1

For customized projects please Contact us:

info@simtrum.com

目录

一、更新记录

版本号	更新内容
V1.5	① 将使用说明书与开发手册合并 ② 增加使用参数 ③ 裸板外形图 ④ 连接说明 ⑤ 使用规格 ⑥ 指令更新 ⑦ 状态码更新
V1.6	① 说明书内容分布修改 ② 通讯指令更新 ③ 脉冲扫描（脉冲输出）功能说明修改 ④ 脉冲控制（脉冲输入）功能说明修改 ⑤ 合并开环使用说明书 ⑥ 外部限位开关接法 ⑦ 软件限位说明增加 ⑧ 合并 DLL 说明 ⑨ 合并更新程序说明书 ⑩ 增加 JC-4xxxC 型号接口说明

二、产品概述

JC-4xxxS,JC-4xxxC（第 1 个 4 表示驱控器，30x 代表本公司 30 系列的压电马达，具体有 301,302,304,308，17x 代表本公司 17 系列的压电马达，具体有 171,172,174，S 跟 C 表示驱控器类型）是一款用于本公司生产的压电马达驱动的精密电动位移台的驱动和控制（以下简称驱控器）。

- 驱控器通过 DB9 接入本公司生产的压电马达，另外通过 DB15 接入位置反馈。
- 驱控器通过 USB 转 485 连接上位机或其他主机。由上位机向驱控器发送基本的运动参数和指令，如 PID 参数，运动速度，确定零位，步进距离，闭环行程，精度等，同时可以实时显示位移台的位置信息。
- 几个驱控器可以串联使用构成多轴（位移台）使用，是一款功能强大，可以灵活使用的驱控装置。
- 闭环控制。
- 开环驱动。
- 支持点位运动，连续运动。支持在线变速、变位功能。
- 支持限位开关。
- 支持脉冲输出模式。
- 支持脉冲输入控制。
- 支持 C，C++，C#等语言调用 DLL 进行控制（配有例程）。
- 支持 labview 调用子 vi 进行控制（配有例程）。
- 支持 Python 控制（配有例程）。

三、驱控器外形尺寸图

图 3.1 驱控器 JC-4xxxS 外形接口图 1

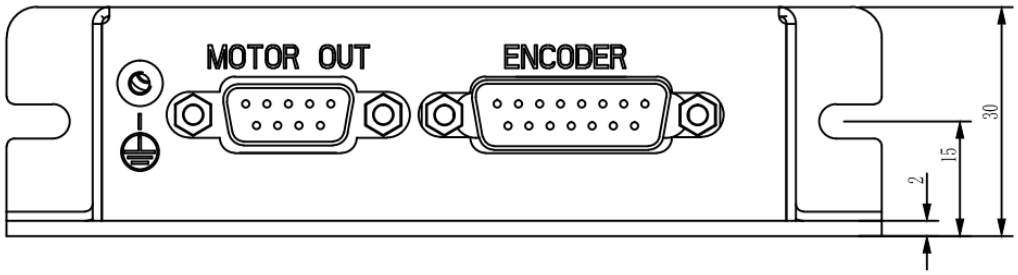


图 3.2 驱控器 JC-4xxxS 外形接口图 2

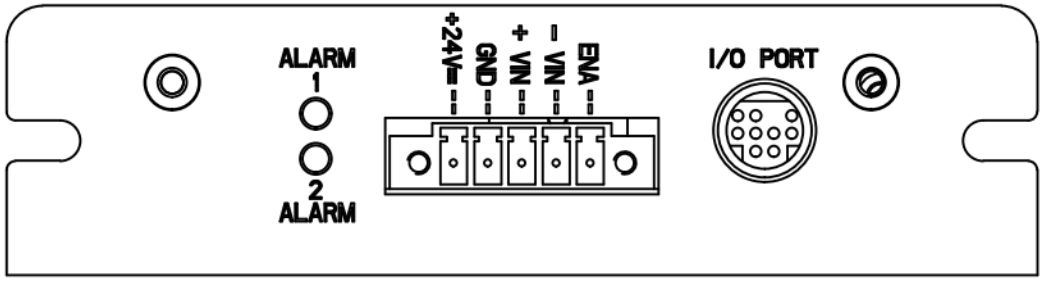
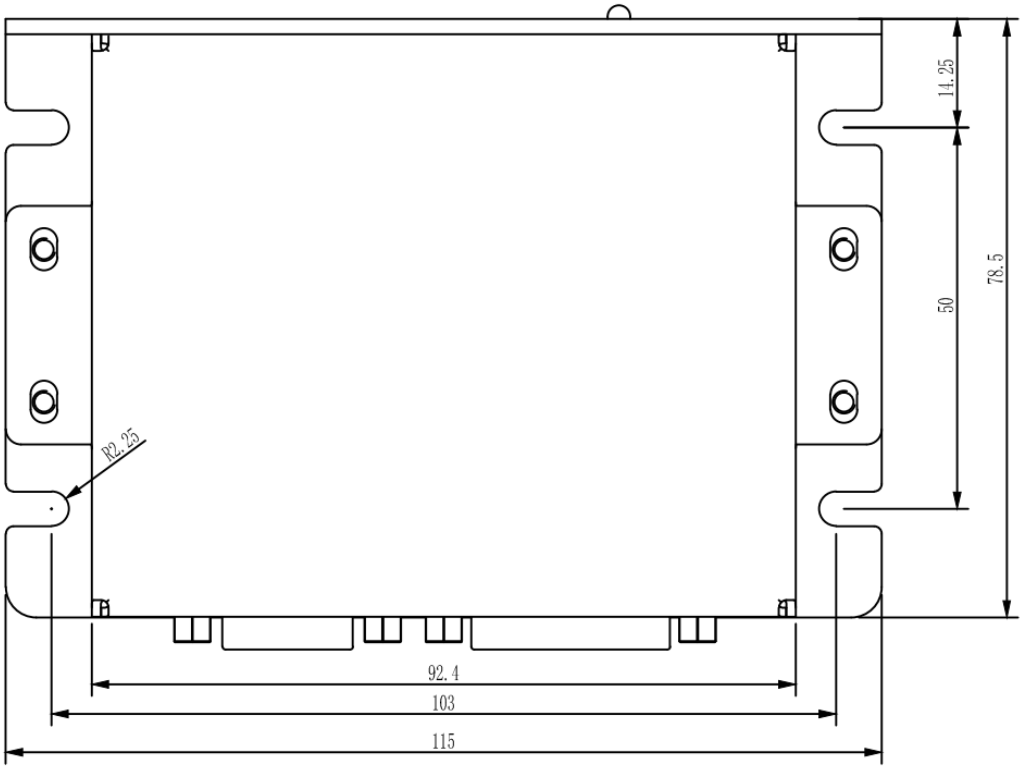


图 3.3 驱控器 JC-4xxxS 外形顶视图



四、驱控器裸板外形图

图 4.1 驱控器 JC-4xxxS 裸板（不带外壳）外形图

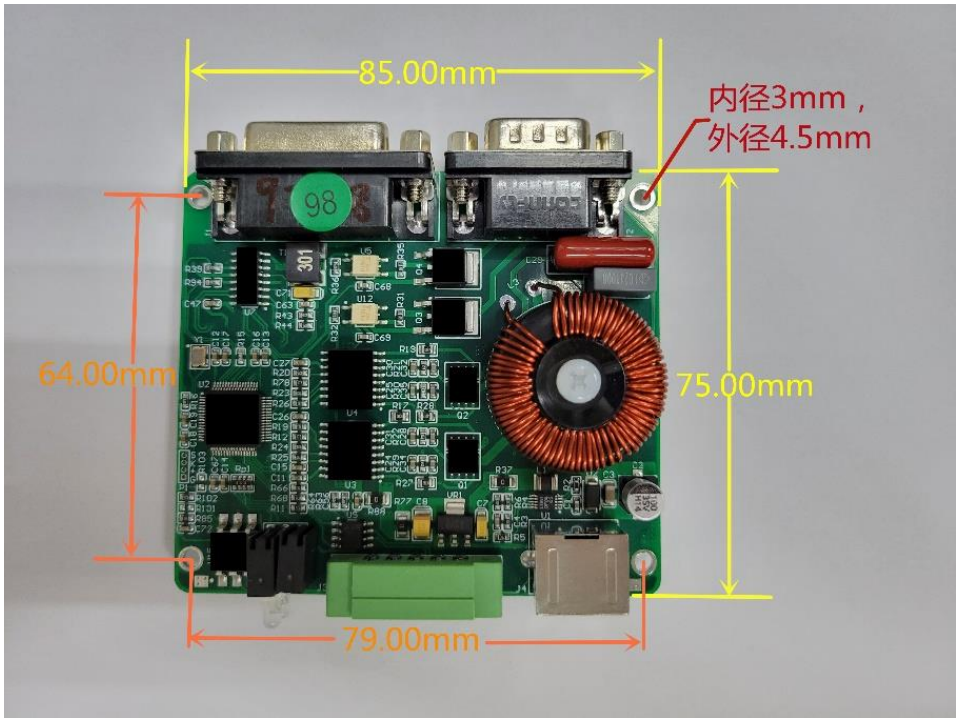
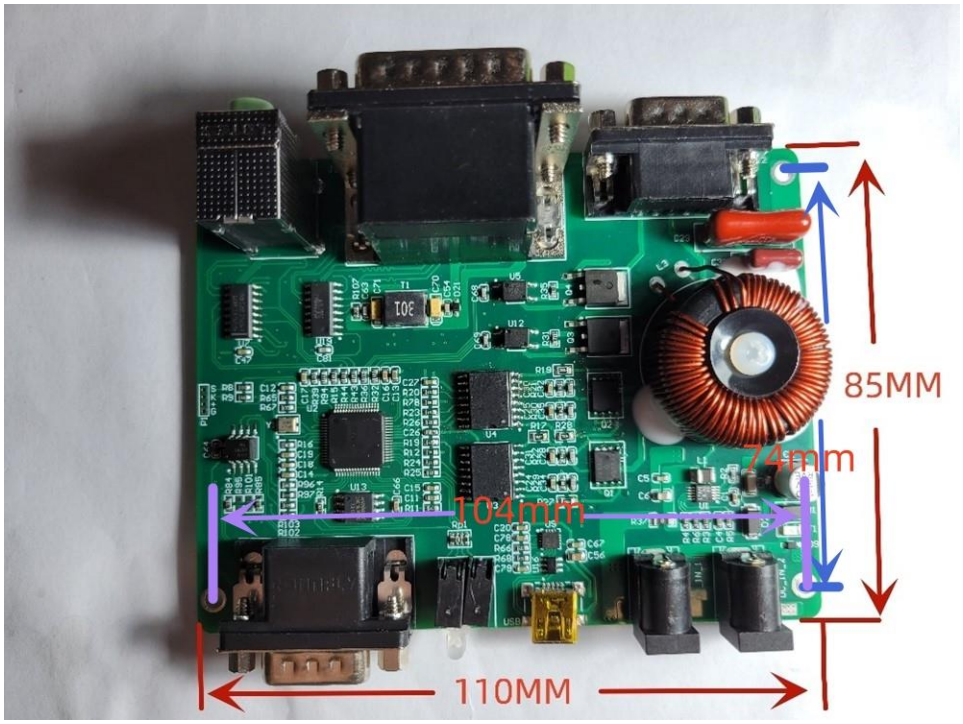


图 4.2 驱控器 JC-4xxxC 裸板（不带外壳）外形图



五、参数

表 5.1 驱控器参数

参数	
使用温度	0 到 40°（支持-40°～70°，需要关闭温度检测）
存储温度	-40～70°
湿度	5～85%，非结露
外部电源（输入）	24V 直流电源，1A（最大输入 30V 直流）
静态功率	<1W（不包括编码器使用功率）
动态功率	<12W

六、硬件连接

6.1 驱控器类型

物品：24V 电源适配器，驱控器，电源/通讯线，位移台，电脑（需安装驱动与控制软件）。

连接步骤简述

（1） 将位移台的马达接口与位置反馈接口分别接入驱控器对应插座。
（2） 将电源/通讯线与驱控器连接。
（3） 使用 USB 通讯线插入电脑。
（4） 将电源适配器与电源/通讯线连接。
（5） 电源适配器接入市电，并上电。

图 6.1 实际连接示例图

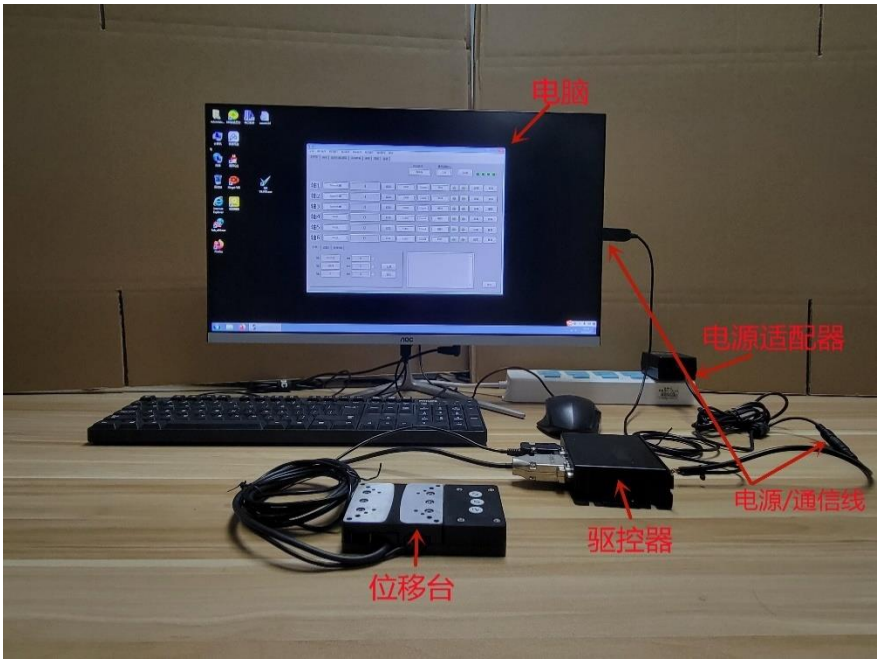
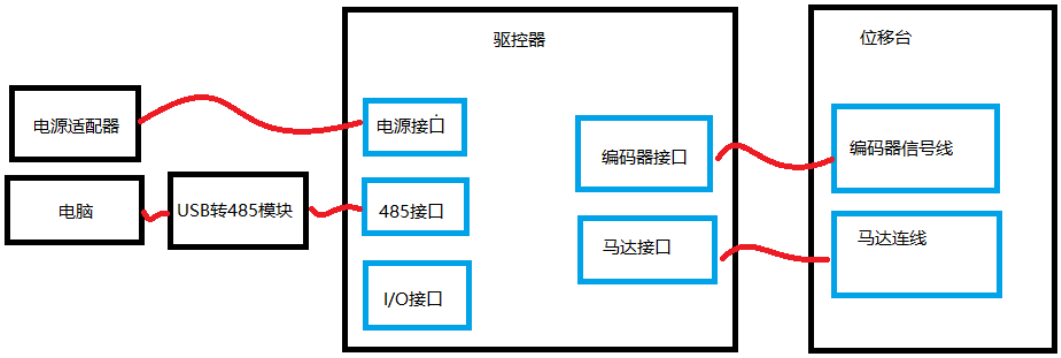


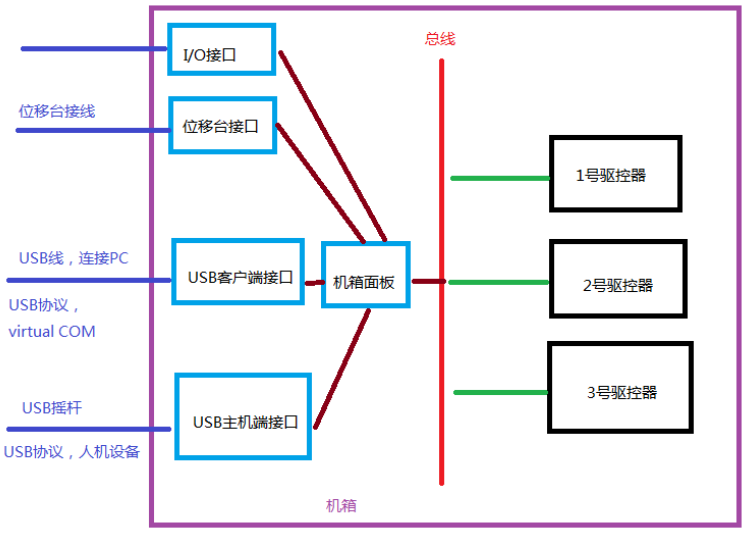
图 6.2 硬件连接框架图



6.2 机箱类型

- 物品：24V 电源适配器，驱控箱，USB 通讯线，位移台，电脑（需安装驱动与控制软件）。
- 机箱存在不同接口，具体连接方式查看相关机箱说明书。

6.3 机箱基本框架图



七、应用程序与使用

需要先安装驱动与上位机。具体查看相关应用程序说明书。
Labview 版本应用程序查看《labview 版本控制软件使用说明书》。

八、驱控器 JC-4xxxS 接口

图 8.1 驱控器 JC-4xxxS 接口图 1

图 8.2 驱控器 JC-4xxxS 接口图 2

8.1 接地螺孔

驱控器外壳接地螺孔。

8.2 电源及通信接口

驱控器的电源接口与 485 通讯接口为 5P，间距 5.08mm 的绿色端子。

表 8.1 电源及通信接口定义

+24V	电源（输入电压范围：20V~30V）
GND	地
+VIN（A）	485 芯片接收器的输入端与驱动器的输出端
-VIN（B）	485 芯片接收器的输入端与驱动器的输出端
ENA	NULL

8.3 指示灯

指示灯由 ALARM1 与 ALARM2 组成，用于显示驱控器的状态。

表 8.2 指示灯的含义

指示灯	绿闪烁	绿常亮	红闪烁	红常亮
ALARM1	找零过程	驱动使能		阻塞保护
ALARM2	马达接入		闪烁一次：通讯输入参数错误	
			1hz 频率闪烁：BootLoader 模式中提示无 APP	马达未接
			2hz 频率闪烁：BootLoader 模式	
			10hz 频率闪烁：BootLoader 模式中提示无秘钥，无法进入 APP	

8.4 I/O 接口

此接口使用的是 PS2 母座（9 芯），孔位编号从右到左递增，从下往上递增。其具体定义如下表 8.3。采用光耦结构，共阳输入。

图 8.3　I/O 接口图

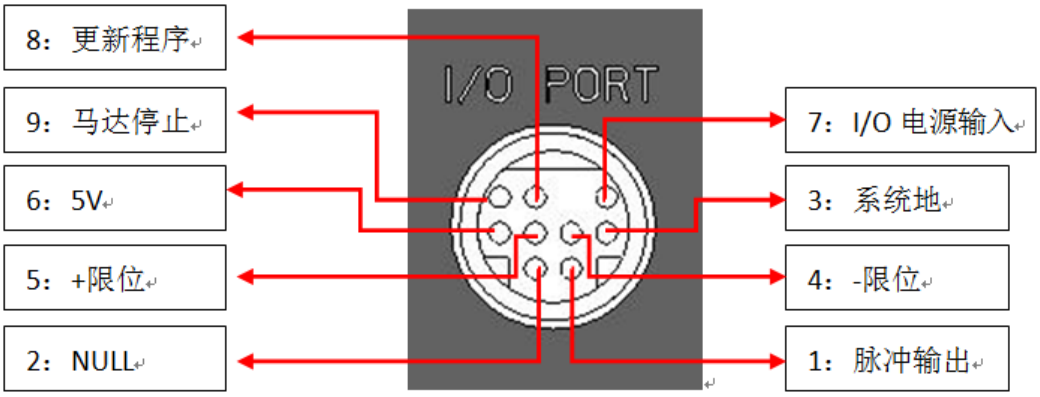


表 8.3　I/O 接口定义

孔号	定义	说明
1	脉冲输出（默认）	脉冲输出引脚（最大电流 30mA）
2	NULL	
3	驱控器 GND	GND
4	-限位	限位检测引脚（最大电流 30mA）
5	+限位	限位检测引脚（最大电流 30mA）
6	5V	输出 5V 电源（最大输出 1A）
7	I/O 电源输入	I/O 功能引脚使用的电源由此电源提供（5~30V）
8	更新程序	更新程序时，此引脚接地

9	马达停止	紧急停止马达（最大电流 30mA）
说明：此 PS2 接口引出接线时请防止接错线引起的短路问题，针脚间短路可能会烧毁驱控器。		

8.5 马达接口

此接口使用的 DB9 两排公座，其具体定义如下表

表 8.4 马达接口定义

DB9 孔号	定义	说明
1	接 GND	接地
2	NULL	
3	相位 A	接马达
4	COM（公共端）	接马达
5	相位 B	接马达
6	检测	DB9 接入检测
7	系统地	
8	NULL	
9	NULL	
说明：6、7 引脚短接，用于检测 DB9 接入		

注意：接插线时请断掉电源，平台驱动时请不要长时间阻碍运动，堵转情况下持续驱动平台可能烧坏电机。

8.6 光栅接口

此接口使用的是 DB15 两排母座，其具体定义如下表。位移反馈通讯协议为正交编码信号，硬件电路为 422 电路。

表 8.5 光栅接口

DB15 接孔号	定义
1	NULL
2	0V
3	NULL
4	Z-
5	B-
6	A-
7	5V
8	5V
9	0V
10	NULL

11	NULL
12	Z+
13	B+
14	A+
15	NULL

九、驱控器 JC-4xxxC 接口

图 9.1 驱控器 JC-4xxxC 接口图 1

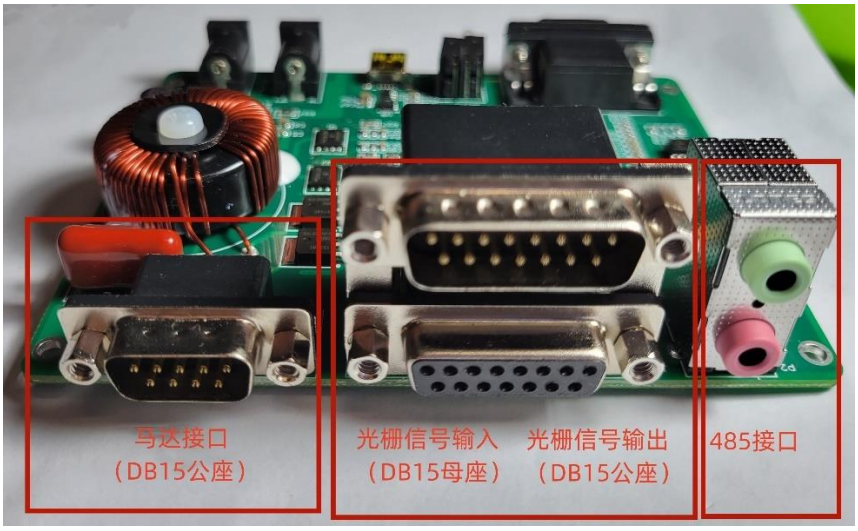
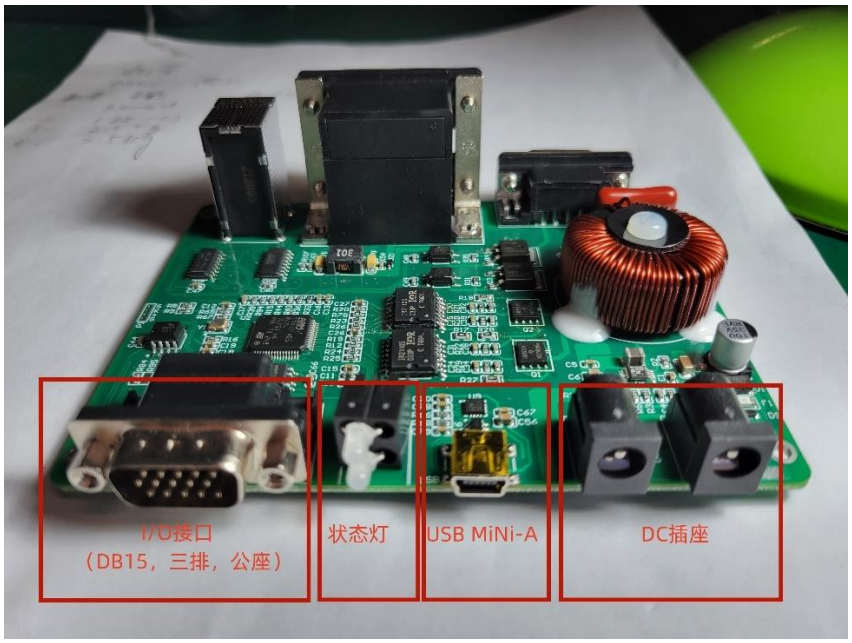


图 9.2 驱控器 JC-4xxxC 接口图 2



9.1 接地孔

驱控器接地位于靠近马达的安装孔（如需断掉，可拆除 R71）。
注意：其他版本的电阻位号不同。

9.2 电源接口

- 驱控器的电源接口为 DC 电源插座（触点外径 6.4mm，触点内径 2mm）。
- 两个都可接入，设计两个 DC 插座是为用于多轴运用（鞠拓结构）。
- 输入电压范围为 20V~30V，建议使用 24V/1A。

9.3 通信接口

驱控器的通讯接口有两种，一种是 485 接口，另一种是 USB 接口（USB mini-A）。485 接口一种是使用双座耳机插座，用于同种驱控器多轴使用；另一种是 I/O 接口（DB15，3 排，公座）。
USB 通讯芯片使用 CH343，可配置 USB 识别符。

9.4 指示灯

指示灯由 ALARM1 与 ALARM2 组成，用于显示驱控器的状态。

表 9.1 指示灯的含义

指示灯	绿闪烁	绿常亮	红闪烁	红常亮
ALARM1	找零过程	驱动使能		阻塞保护
ALARM2	马达接入		闪烁一次：通讯输入参数错误	
			1hz 频率闪烁：BootLoader 模式中提示无 APP	马达未接
			2hz 频率闪烁：BootLoader 模式	
			10hz 频率闪烁：BootLoader 模式中提示无秘钥，无法进入 APP	

9.5I/O 接口

此接口使用的是 3 排 DB15 公座，。其具体定义如下表 9.2。

表 9.2　I/O 接口定义

孔号	定义	说明
1	I/O 电源输入	I/O 功能引脚使用的电源由此电源提供（5~30V）
2	Dir+	脉冲控制的方向引脚输入
3	Dir-	脉冲控制的方向引脚输出
4	Plu+	脉冲控制的脉冲计数引脚输入
5	Plu-	脉冲控制的脉冲计数引脚输出
6	5V	输出 5V 电源（最大输出 1A）
7	-限位	限位检测引脚（最大电流 30mA）
8	+限位	限位检测引脚（最大电流 30mA）
9	马达停止	紧急停止马达（最大电流 30mA）
10	更新程序	更新程序时，此引脚接地
11	脉冲输出	脉冲输出引脚（最大电流 30mA）
12	NULL	
13	NULL	
14	NULL	
15	GND	驱控器 GND
说明：此接口引出接线时请防止接错线引起的短路问题，针脚间短路可能会烧毁驱控器。		

9.6 马达接口

此接口使用的 DB9 两排公座，其具体定义如下表

表 9.3 马达接口定义

DB9 接孔号	定义	说明
1	接 GND	接地
2	NULL	
3	相位 A	接马达
4	COM（公共端）	接马达
5	相位 B	接马达
6	检测	DB9 接入检测
7	系统地	
8	NULL	
9	NULL	
说明：6、7 引脚短接，用于检测 DB9 接入		

注意：接插线时请断掉电源，平台驱动时请不要长时间阻碍运动，堵转情况下持

续驱动平台可能烧坏电机。

9.7 光栅接口

此接口使用的是双层两排 DB15，上公下母，其具体定义如下表。光栅信号输入接入母座，光栅信号输出为公座。位移反馈通讯协议为正交编码信号，硬件电路为 422 电路。

9.4 光栅接口

DB15 接孔号	定义
1	NULL
2	0V
3	NULL
4	Z-
5	B-
6	A-
7	5V
8	5V
9	0V
10	NULL
11	NULL
12	Z+
13	B+
14	A+
15	NULL

十、开发

10.1 使用 window 编写应用程序进行控制

- 使用 C，C++，C#等语言编写应用程序，可调用 DLL 相关函数进行对驱动器连接，设置参数，定位控制等。相关函数具体用法查看“动态链接库说明”。可参考相关例程。
- 使用 labview 开发，可调用相关子 vi 进行编写应用程序。可参考相关例程。
- 使用 Python 开发，参考相关例程。

10.2 使用单片机，PLC，DSP 等 MCU 控制

相关串口配置参数与指令查看“通信格式及意义”与对应功能篇章。

十一、通信格式及意义

11.1 串口配置参数

- 波特率为：115200。
- 数据位为：8。
- 停止位为：1 。

11.2 格式说明

1 条指令含有 10 个字节，起始字节编号为 0，结束字节编号为 9，具体含义如下表所示。

表 11.1 数据包格式

含义	起始	数据源	地址	数据类型	数据区	状态	校验和
字节编号	0	1	2	3	4、5、6、7	8	9
大小	1byte	1byte	1byte	1byte	4byte	1byte	1byte
数值	0xA5	0x58('X')					
		0x53('S')					

通信格式说明：

1） 起始位： 固定是 0xA5；

2） 数据源： 0x58('X')代表下位机发送给上位机，
0x53('S')代表上位机发送给下位机。

3） 地址： 上下位机建立通信的地址号，地址范围 0x01-0xFE，地址 0x00 为中继通信设备保留，0xFF 为广播地址（此地址所有除中继设备之外的设备都可以响应）。

4） 数据类型： 标记了数据区的数据类型、格式，上位机、下位机使用的数据类型相同。

5） 数据区： 具体通信的内容，长度依据数据类型的不同而不同。

6） 状态： 标示运行状态信息，或控制信息。

7） 校验和： 除起始 2 字节外其他数据相加结果取后 8 位为校验和数据。

11.3 串口指令集

表 11.2 上位机给下位机发送的数据格式（蓝色字为下位机返回数据含义）

数据类型 Byte3	数据区 byte4～byte7 含义	读写属性
0x81	0x5445:询问误差 0x5450:查询位置坐标	RO
0x82	绝对定位	WO
0x83	相对定位	WO
0x8E	0x4C('L') : "-" 向 JOG 模式 0x52('R'): "+" 向 JOG 模式 0x53('S'): 停止 JOG 模式	WO
0x7C	0x53('S'):停止运动	WO
0x54	查询运行状态	RO
0xA1	查询硬件/软件版本	RO
0x5F	查询 BootLoader 版本	RO
0x40	机器流水码	RO
0x4E	查询温度	RO
0x60	0：位置坐标正相 1：位置坐标反相	RW
0x57	0:关闭碰撞检测保护 7:启动碰撞检测保护	RW
0x59	设置碰撞系数	RW
0x85	定位速度,单位（×10）counts/ms,输入范围 2～100 (推荐使用 0x7A，此为旧版本)	RW
0x7A	定位速度,单位 counts/s,输入范围 20000～1000000	RW
0x9D	JOG 速度,单位（×10）counts/ms,输入范围 2～100	RW
0x7B	JOG 速度,单位 counts/s,输入范围 20000～1000000	RW
0x9B	找零速度,单位（×10）counts/ms,输入范围 2～100	RW

0x79	找零速度,单位 counts/s,输入范围 20000～1000000	RW
0x9E	扫描速度,单位（×10）counts/ms,输入范围 2～100	RW
0x5A	加速度与减加速度，高 16 位为加速度，低 16 位为减加速度，单位：（Accel÷100）counts/ms^2	RW
0x96	调节周期,单位 100us,输入范围 1～10	RW
0x84	0x57:参数存储	WO
0xA0	0x55AA55AA:复位	W0
0x8B	0x5959:设置开机找零（反向） 0x59:设置开机找零（正向） 0x2B:启动正向找零（单次） 0x2D:启动负向找零（单次） 0x4E:取消找零	RW
0x92	开机零位偏移量	RW
0x8A	置零	RO
0x5D	找零碰撞上限次数	RW
0x6E	开机延迟找零时间,单位 ms,输入范围 0～50000	RW
0x87	最小边界点	RW
0x88	最大边界点	RW
0x51	起始 PWM,输入范围 150～350	RW
0x8F	比例量,不建议修改	RW
0x90	积分量,不建议修改	RW
0x91	微分量,不建议修改	RW
0x48	惯性系数,不建议修改	RW
0x6D	波特率,输入范围 2400～256000	RW
0x58	重置点	RW

0x8C	设置轴号,输入范围 1～8	RW
0x95	闭环细调模式起始 PWM，输入范围 150～350	RW
0x45	闭环细调模式范围，输入范围 0～100	RW
0x46	闭环细调模式脉宽，单位 10us，输入范围 10～50	RW
0x5B	闭环细调模式递增值，输入范围 1～20	RW
0x61	开环细调档位频率，单位 0.01hz， 输入范围 50～20000（0.5～2Khz）	RW
0x62	开环细调档位脉宽，单位 10us，输入范围 1～50	RW
0x63	开环细调档位幅值，输入范围 1～350	RW
0x68	开环细调档位启动与脉冲数，±代表方向	WO
0xA8	开环粗调档位设置	WO

注意：

1、以上可写参数修改属于临时修改，如需设置为开机默认参数，则需要保存到 flash（使用存储指令）。

2、0 表示 0x00。

3、对于可读数据，查询时，需要将状态位置 1（只读参数可不置 1）。

4、下位机反馈给上位机的数据类型为 0x50（参考错误（异常）反馈机制说明）

5、下位机反馈给上位机数据中的数据区 Byte4 为高位字节，Byte7 为低位字节，Byte4、Byte5、Byte6、Byte7 组成 32bit 数据（除特殊说明外，数据类型一般为：有符号字，signed int）。

6、除了特殊说明的反馈信息，其他的如上一条所示。

例 1 位置查询

标号	0	1	2	3	4	5	6	7	8	9
发送	0xA5	0x53	0x01	0x81	0x00	0x00	0x54	0x50	0x00	0x26
接收	0Xa5	0x58	0x01	0x81	*	*	*	*	*	*

注： *代表根据实际情况自动变化的数字

例 2 绝对坐标运动

运动到坐标 0x1234（4660）处

标号	0	1	2	3	4	5	6	7	8	9
发送	0xA5	0x53	0x01	0x82	0x00	0x00	0x12	0x34	0x01	0xCA
接收	0xA5	0x58	0x01	*	*	*	*	*	*	*

11.4 状态位

本章对通信过程中的状态字节做详细说明。

表 11.3， 状态字节在通信数据包中的位置

含义	起始	数据源	地址	数据类型	数据区	状态	较验和
字节编号	0	1	2	3	4、5、6、7	8	9
大小	1byte	1byte	1byte	1byte	4byte	1byte	1byte
数值	0xA5	0x58('X')					

表 11.4，上位机发送的状态字节每一位含义

上位机发送的状态字节								
位号	7	6	5	4	3	2	1	0
位状态							查询参数	驱动使能
1							查询	开
0							不查询	关

图 11.1 上位机发送的状态字节含义图

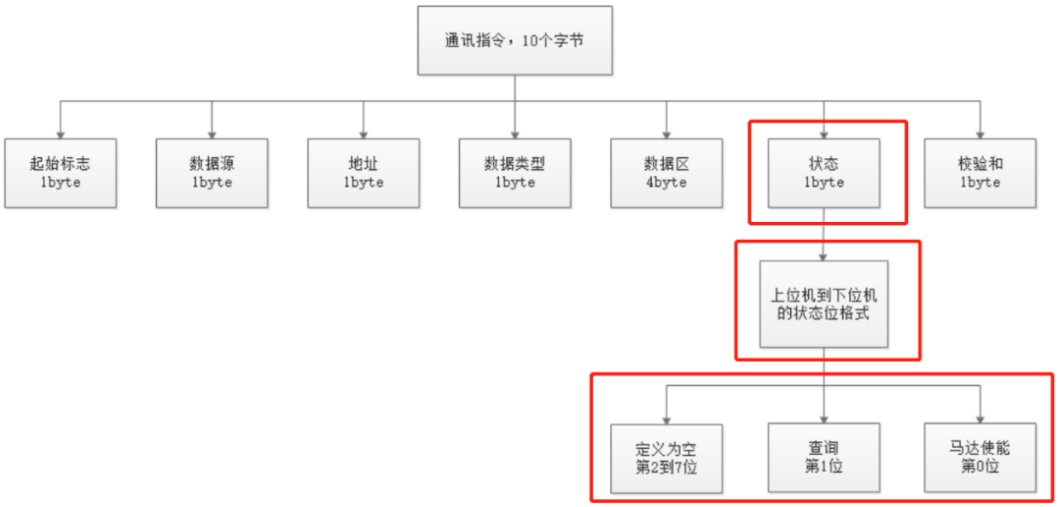
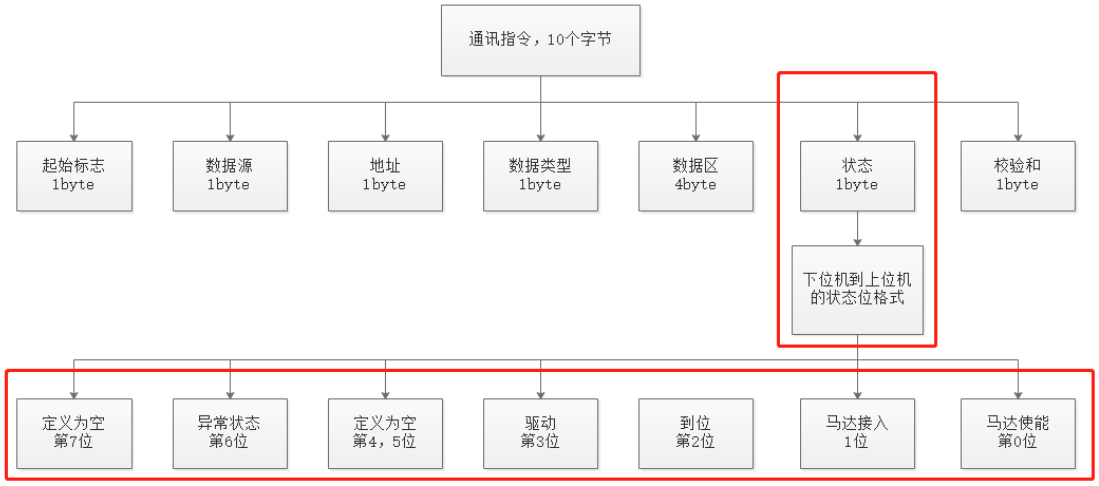


表 11.5 下位机发送的状态字节每一位含义

下位机发送的状态字节								
位号	7 位	6 位	5 位	4 位	3 位	2 位	1 位	0 位
位状态	/	异常（详情查看运行异常）	/	/	驱动	到位	马达接入	驱动使能
1		是			是	是	是	开
0		否			否	否	否	关

注意：“到位”是指位移台定位完成。位移台未驱动并且位移台未到位的现象存在于，位移台出现异常。

图 11.2 下位机发送的状态字节含义图

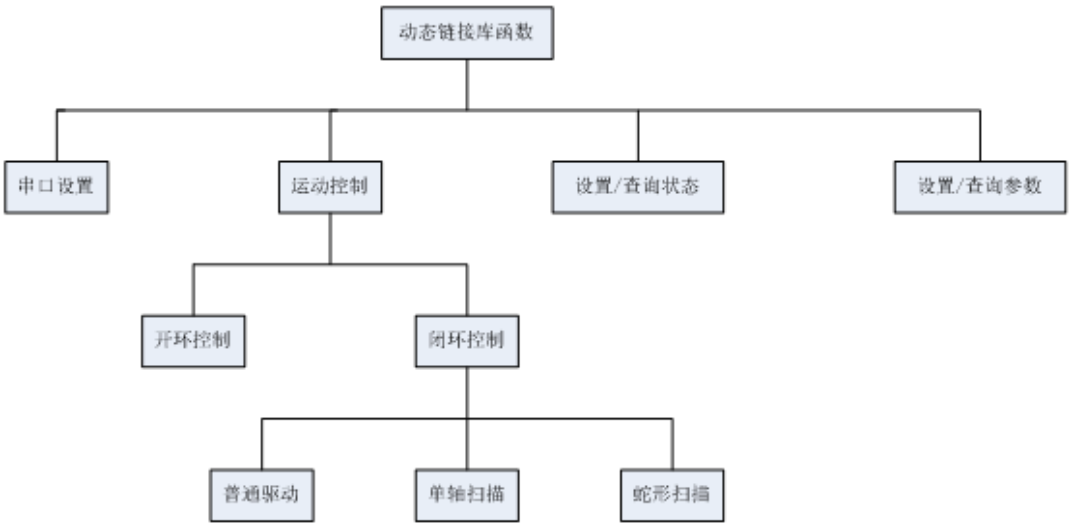


十二、动态链接库说明

文件名为：SeriolCom.dll。用于 Windows 系统开发控制软件使用，提供 32 位与 64 位版本。

12.1 函数功能分布框架

图 12.1 DLL 函数框架图



如图所示，动态链接库函数可分成四个功能模块的函数。

12.2DLL 函数集

表 12.1　DLL 函数表

函数名	功能
DLL_OpenCom	打开通信串口
DLL_CloseCom	关闭通信串口
DLL_OpenQK	自动连接设备
DLL_SetPpdatePeriod	设置数据更新周期
DLL_GetPpdatePeriod	查询数据更新周期
GetDLLVersion	查询 DLL 版本号
DLL_GetError	读取最近一次软件配置出错的错误信息
SetControlAxis	设置可操作轴
DLL_ReadPosition	读取指定轴的位置
DLL_AxisEnable	控制器使能与失能设置
DLL_GetAxisStatus	查询驱控器状态
DLL_PositionRelativeMove	相对移动

DLL_PositionAbsoluteMove	绝对移动
DLL_JOG_Move	JOG 移动
DLL_SetSpeed4	速度设置
DLL_GetSpeed4	查询速度
DLL_SetAccelValue	设置加速度
DLL_GetAccelValue	查询加速度
DLL_Mod2ScanControl2	单轴扫描
DLL_Mod2ScanStartPosition	设置单轴扫描的起始点
DLL_Mod2ScanStartPulseSet	设置单轴扫描的脉冲触发间距和 脉冲触发数量
DLL_Mod2ScanPulseWidthSet	设置单轴扫描的脉宽
DLL_Mod2ScanMaskPulseSet	设置单轴扫描脉冲掩盖的数量
TAS_control	蛇形扫描控制
TAS_scanRangeSet	设置蛇形扫描的参数
TAS_getStatus	读取蛇形扫描状态信息
DLL_PulseTrigger	设置到位触发脉冲
DLL_GetStatusError	查询控制器运动状态
DLL_ClearMotorError	清除驱控器的异常状态
DLL_SetZeroPosition	设置位移台当前位置为零点
DLL_SetFIMode_S	设置开机寻零
DLL_GetFIMode_S	查询开机寻零模式
DLL_ChangeAxisNum	设置驱控器轴号
DLL_GetPipelineCode	查询机器流水码
DLL_AskVersionIum	查询驱控器版本
DLL_SetBaseValue	设置基础参数
DLL_GetBaseValue	查询基础参数
DLL_SetBoundary	设置最大/最小边界点
DLL_GetBoundary	查询最大/最小边界点
DLL_SetStepValue	设置细调模式参数
DLL_GetStepValue	查询细调模式参数
DLL_SetEPW	保存驱控器参数
CoarseOpenDrive	开环驱动的粗调控制
DLL_FineOpenDrive	开环驱动的细调档位控制
DLL_SetFineOpenFre	设置开环驱动的细调档位的频率
DLL_SetFineOpenWidth	设置开环驱动的细调档位的脉宽
DLL_SetFineOpenAmp	设置开环驱动的细调档位的幅值
说明：带红色标注函数为需使用项	

12.3 串口设置相关的函数

这类函数主要有打开串口函数、关闭串口函数以及设置通信周期函数等函数。动态链接库的大多数函数都需要在打开串口后才能正常使用。

打开串口方式有 3 种，这三种差异性如下：

- DLL_OpenCom 用于通过输入参数连接对应 COM 口；
- DLL_OpenQK 可用于自动连接 1 到 2 个设备；
- DLL_OpenQK 可用于连接特定 USB 标识符的 COM 口。

函数名	DLL_OpenCom		
函数原型	int DLL_OpenCom(int Com,int Baud)		
函数功能	打开通信串口		
	变量名	变量类型	说明
形参	Com	有符号 32 位整型	串口号（1,2,3…）
	Baud	有符号 32 位整型	波特率设置（默认使用 115200）
返回值		有符号 32 位整型	串口成功打开，返回 1，否则返回-1
说明： 1、此函数只打开一个可通信串口，无法通过调用此函数两次来打开两个可通信串口。 2、此函数保留，用于兼容旧版本			

函数名	DLL_CloseCom		
函数原型	int DLL_CloseCom(void)		
函数功能	关闭通信串口		
	变量名	变量类型	说明
形参	无		
返回值		有符号 32 位整型	串口成功关闭或者串口本身处于关闭状态，返回 1，否则返回-1
说明：此函数只与 DLL_OpenCom 函数配合使用			

函数名	DLL_OpenQK		
函数原型	int DLL_OpenQK(int controlNum)		
函数功能	自动连接设备		
	变量名	变量类型	说明
形参	controlNum	有符号 32 位整型	输入‘0’关闭连接， 输入‘2’连接设备（单串口）， 输入‘4’连接设备（单串口，识别特定 USB 标志符）， 输入‘5’连接设备（双串口）
返回值		有符号 32 位整型	设置成功返回 1，否则返回-1
说明： 1、 默认使用的波特率为 115200； 2、 用于自动连接，除了‘4’，自动连接原理是通过对计算机上所有串口发送指令，查询有正确反馈的串口，则为对应串口； 3、 ‘4’的自动连接原理是通过查询 USB 标识符，识别出标识符为 82517200 的串口，此方式需使用指定的串口设备； 4、 可通过设置‘5’，最大连接两个串口使用，此时将引入映射概念。			

函数名	DLL_SetPpdatePeriod		
函数原型	int DLL_SetPpdatePeriod(unsigned int period)		
函数功能	设置数据更新周期		
	变量名	变量类型	说明
形参	period	无符号 32 位整型	输入范围为[10,200]，单位 ms
返回值		有符号 32 位整型	设置成功返回 1，设置失败返回-1
说明：此函数的功能为设置数据更新周期，需咨询工程师，请谨慎设置。			

函数名	DLL_GetPpdatePeriod		
函数原型	unsigned int DLL_GetPpdatePeriod(void)		
函数功能	查询数据更新周期		
	变量名	变量类型	说明
形参	无		
返回值		无符号 32 位整型	返回更新周期，单位 ms

函数名	GetDLLVersion		
函数原型	unsigned int GetDLLVersion (void)		
函数功能	查询 DLL 版本		
	变量名	变量类型	说明
形参	无		
返回值		无符号 32 位整型	返回 DLL 版本

函数名	DLL_GetError		
函数原型	unsigned int DLL_GetError(void)		
函数功能	读取最近一次软件配置出错的错误信息		
	变量名	变量类型	说明
形参	无		
返回值		无符号32位整型	错误信息码含义如下所示： 返回 0：没有错误； 返回 991：马达相关错误； 返回 992：没有检测到马达连接； 返回 993：没有使能马达； 返回 994：没有连接到设备； 返回 995：查询的目标轴位置没有最新内容 返回 996：输入参数不符合要求 返回 997：DLL_OpenCom 打开失败 返回 998：DLL_CloseCom 关闭失败 返回 999：串口处于关闭状态 返回 1000：控制器收到指令，但没有正常设置（可能由于控制器处于异常状态或者设置内容与其他设置项冲突） 返回 1001：设置超时 返回 1002：Socket 初始化失败 返回 1003：Socket 链接失败 返回 1004：未知 Socket 返回 1100：操作轴号无效

12.4 运动控制部分的函数

- 这类函数可分为开环控制与闭环控制两大分类，
- 闭环控制部分除了普通驱动函数，还有单轴扫描相关函数和蛇形扫描相关函数。
- 在进行运动控制之前，不仅要打开串口确保通讯正常，还必须先使能轴号与马达。具体函数查看相关的功能。

函数名	SetControlAxis		
函数原型	void SetControlAxis(unsigned int Axis_Shift)		
函数功能	使能可操作的驱控器的标志位		
	变量名	变量类型	说明
形参	Axis_Shift	无符号 32 位整型	设置可操作轴号标志位， 轴一 0x01（第 0 位为 1）， 轴二 0x02（第 1 位为 1）， 轴三 0x04（第 2 位为 1）， 轴 1 跟轴 2 同时生效为 0x03 （第 0 跟第 1 位为 1） 以此类推，
返回值	空		

函数名	DLL_ReadPosition		
函数原型	int DLL_ReadPosition(unsigned int Address)		
函数功能	读取指定轴的位置		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
返回值		有符号 32 位整型	返回位置信息

函数名	DLL_AxisEnable		
函数原型	int DLL_AxisEnable (unsigned int Address, unsigned char KG)		
函数功能	控制器使能与失能设置		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	KG	字符型	输入“K”设置马达使能， 输入“G”设置马达失能
返回值		有符号 32 位整型	设置成功返回 1，否则返回-1
说明：设置失败需查询是否已连接上驱控器。			

函数名	DLL_GetAxisStatus		
函数原型	int DLL_GetAxisStatus (unsigned int Address, unsigned int StatusNum)		
函数功能	查询驱控器状态		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	StatusNum	无符号 32 位整型	输入“1”查询驱控器是否使能， 输入“2”查询是否运动， 输入“3”查询马达是否接入， 输入“4”查询是否有异常反馈。
返回值		有符号 32 位整型	返回 1 则状态位“是”， 返回 0 则状态位“否”， 返回-1 则读取失败。
说明：用于判断驱控器是否正常定位，使能状态等			

函数名	DLL_PositionRelativeMove		
函数原型	int DLL_PositionRelativeMove (unsigned int Address, int Position)		
函数功能	相对移动（以当前平台当前位置为参考进行移动）		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Position	有符号 32 位整型	位移距离设置值，正负则决定位移的方向，单位为 counts， 即位置反馈的分辨率
返回值		有符号 32 位整型	返回 1 则指令发送成功， 返回-1 则指令发送异常
说明：注意使能马达			

函数名	DLL_PositionAbsoluteMove		
函数原型	int DLL_PositionAbsoluteMove (unsigned int Address, int Position)		
函数功能	绝对移动（以所设零点位置为参考进行移动）		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Position	有符号 32 位整型	坐标设置值
返回值		有符号 32 位整型	返回 1 则指令发送成功， 返回-1 则指令发送异常
说明：注意使能马达			

函数名	DLL_JOG_Move		
函数原型	int DLL_JOG_Move (unsigned int Address, unsigned char CMD)		
函数功能	JOG 模式移动		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	CMD	字符型	输入“L”正方向连续移动， 输入“R”右方向连续移动， 输入“T”停止移动。
返回值		有符号 32 位整型	返回 1 则指令发送成功， 返回-1 则指令发送异常。

函数名	DLL_GetStatusError		
函数原型	unsigned int DLL_GetStatusError (unsigned int Address, unsigned char* Output)		
函数功能	查询驱控器的运动状态		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	查询驱控器的轴号（1,2,3...）
	Output	字符指针	保存状态异常类型的说明
返回值		无符号 32 位整型	返回 0 则状态正常， 返回其他正整数为异常状态码， 返回-1 则读取失败。

函数名	DLL_ClearMotorError		
函数原型	int DLL_ClearMotorError(unsigned int Address)		
函数功能	查清除驱控器的异常状态		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	查询驱控器的轴号（1,2,3...）
返回值		有符号 32 位整型	返回 1 则设置成功， 返回-1 则设置失败
说明：当驱动异常触发时，驱控器会标记异常，此时该轴的驱控器会受到保护，无法响应运动控制指令。此函数用于清除该轴的异常标记，使该函数能再次响应运动控制指令。但是未解决异常原因的情况下，再次驱动往往会再出现异常，可通过 DLL_GetStatusError 函数找出异常原因。			

函数名	DLL_SetZeroPosition		
函数原型	int DLL_SetZeroPosition(unsigned int Address)		
函数功能	设置位移台当前位置为零点		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
返回值		有符号 32 位整型	返回 1 设置成功， 返回-1 设置失败。

函数名	DLL_SetSpeed4		
函数原型	int DLL_SetSpeed4(unsigned int Address, unsigned char Type, unsigned int countsPerS)		
函数功能	速度设置（最低分辨率为 10000counts/s，速度设置范围 30000~600000counts/s，其他范围需咨询工程师）		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Type	字符型	输入“V”设置普通定位速度， 输入“J”设置 JOG 位移速度， 输入“F”设置开机找零位速度， 输入“S”设置扫描速度。
	countsPerS	无符号 32 位整型	速度设置值，单位 counts/s
返回值		有符号 32 位整型	返回 1 设置成功， 返回-1 设置失败。

函数名	DLL_GetSpeed4		
函数原型	int DLL_GetSpeed4(unsigned int Address, unsigned char Type)		
函数功能	读取设置速度（counts/s）		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Type	字符型	输入“V”查询普通定位速度， 输入“J”查询 JOG 位移速度， 输入“F”查询开机找零位速度， 输入“S”查询扫描速度
返回值		有符号 32 位整型	读取成功则返回速度值， 读取失败返回-1。

函数名	DLL_SetAccelValue		
函数原型	int DLL_SetAccelValue(unsigned int Address, unsigned int Accel, unsigned int Decel)		
函数功能	设置加速度		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Accel	无符号 32 位整型	加速加速度，单位：（Accel÷100）counts/ms^2
	Decel	无符号 32 位整型	减速加速度，单位：（Accel÷100）counts/ms^2
返回值		有符号 32 位整型	返回 1 则设置成功， 返回 0 则没有收到返回， 返回-1 则设置失败。
说明：谨慎设置。			

函数名	DLL_GetAccelValue		
函数原型	int DLL_GetAccelValue (unsigned int Address, unsigned int Mode)		
函数功能	查询加速度		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Mode	无符号 32 位整型	Mode 为 1 时查询加速加速度， Mode 为 2 时查询减速加速度
返回值		有符号 32 位整型	查询成功时返回 Mode 对应的参数的值，查询失败时返回-1

单轴扫描与蛇形扫描相关 DLL 查看扫描触发模式（脉冲输出）章节。

12.5 设置驱控器参数相关的函数

函数名	DLL_SetBaseValue		
函数原型	int DLL_SetBaseValue (unsigned int Address, int Value, unsigned int Mode)		
函数功能	设置基础参数		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Value	有符号 32 位整型	对应参数的值
	Mode	无符号 32 位整型	Mode 为 1 设置标记点偏移量； Mode 为 2 设置标记点误差范围； Mode 为 3 设置上电延迟找零时间，单位：ms； Mode 为 4 设置碰撞检测使能（设置 0 为关闭，设置 1 为开启）； Mode 为 5 设置起始 PWM 值，输入范围 150～350； Mode 为 6 设置重置点； Mode 为 7 设置参考方向（设置 0 为正向显示，设置 1 为反向显示）； Mode 为 8 设置调速周期； Mode 为 9 设置碰撞系数； Mode 为 10 设置反馈信号精度。
返回值		有符号 32 位整型	返回 1 设置成功， 返回 0 没有收到返回数据， 返回-1 设置失败。

函数名	DLL_GetBaseValue		
函数原型	int DLL_GetBaseValue (unsigned int Address, unsigned int Mode)		
函数功能	查询步进参数		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Mode	无符号 32 位整型	Mode 为 1 查询标记点偏移量， Mode 为 2 查询标记点误差范围， Mode 为 3 查询上电延迟找零时间， Mode 为 4 查询边界检测开关（返回 0 为关闭，返回 1 为开启）， Mode 为 5 查询起始 PWM 值， Mode 为 6 查询重置点， Mode 为 7 查询参考方向（返回 0 为正向显示，返回 1 为反向显示）， Mode 为 8 查询调速周期，

			Mode 为 8 查询调速周期， Mode 为 9 查询碰撞系数， Mode 为 10 查询反馈信号精度
返回值		有符号 32 位整型	查询成功时返回 Mode 对应的参数的值，查询失败时返回-1

函数名	DLL_SetStepValue		
函数原型	int DLL_SetStepValue (unsigned int Address, int Value, unsigned int Mode)		
函数功能	设置细调模式参数		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Value	有符号 32 位整型	对应参数的值
	Mode	无符号 32 位整型	Mode 为 1 设置细调模式 PWM，输入范围 150～350； Mode 为 2 设置细条模式驱动时长，单位 us，输入范围 150～500； Mode 为 3 设置细调模式最大 P 值； Mode 为 4 设置细调模式最小 P 值； Mode 为 5 设置细调模式最大驱动时长； Mode 为 6 设置细调模式最小驱动时长； Mode 为 7 设置细调模式调节范围； Mode 为 8 设置细调模式 P 值增长幅度，输入范围 1 到 10； Mode 为 9 设置 AB 相相差时长； Mode 为 10 设置细调模式强制设置范围；
返回值		有符号 32 位整型	返回 1 设置成功， 返回 0 没有收到返回， 返回-1 设置失败。

函数名	DLL_GetStepValue		
函数原型	int DLL_GetStepValue (unsigned int Address, unsigned int Mode)		
函数功能	查询细调模式参数		
	变量名	变量类型	说明
形参	Address	有符号 32 位整型	驱控器的轴号（1,2,3...）
	Mode	有符号 32 位整型	Mode 为 1 查询细调模式 PWM， Mode 为 2 查询细条模式驱动时长， Mode 为 3 查询细调模式最大 P 值，

			Mode 为 4 查询细调模式最小 P 值， Mode 为 5 查询细调模式最大驱动时长， Mode 为 6 查询细调模式最小驱动时长， Mode 为 7 查询细调模式运行范围， Mode 为 8 查询细调模式 P 值增长幅度， Mode 为 9 查询 AB 相相差时长， Mode 为 10 查询细调模式强制范围
返回值		无符号 32 位整型	查询成功时返回 Mode 对应的参数的值，查询失败时返回-1

12.6 开环驱动相关函数

开环驱动相关 DLL 查看开环使用章节。

十三、功能介绍

13.1 定位运动

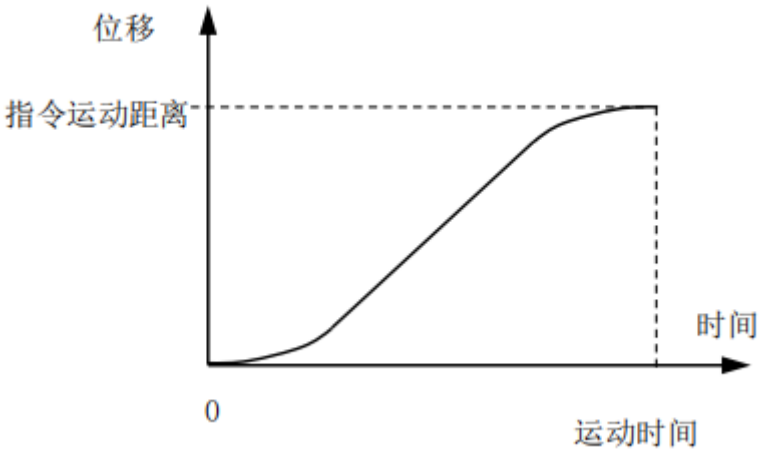
- 定位运动是指：驱控器控制位移台从当前位置开始以设定的速度运动到指定位置后准确地停止。运动轨迹如下图 13.1 所示。具体可查看“速度曲线说明”。
- 定位运动分绝对定位和相对定位。
- 运动过程中发送新的位置指令。

  在进行定位的过程中，当发送新指令到驱控器，则改变定位的终点位置。比如原本绝对定位到 50000；但当运动到了 40000 时，发送绝对定位命令将终点改为 80000，电机从减速变为加速，继续向前运动，然后减速停在新的终点位置。

  比如原本相对定位到 50000；但当运动到了 40000 时，发送相对定位指令相对移动 50000，将终点改为 100000（可配置成已当前点为参考点，则终点改为 90000），电机从减速变为加速，继续向前运动，然后减速停在新的终点位置。
- 运动过程中改变速度

  在定位过程中，改变速度，则根据写入的速度进行重新计算运动曲线。具体可查看“速度曲线说明”。

图 13.1 运动轨迹示意图



13.1.1 绝对定位

以坐标 0 位置为参考移动到目标坐标。

相关 DLL 函数如下所示：

函数名	DLL_PositionAbsoluteMove		
函数原型	int DLL_PositionAbsoluteMove (unsigned int Address, int Position)		
函数功能	绝对移动（以所设零点位置为参考进行移动）		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Position	有符号 32 位整型	坐标设置值
返回值		有符号 32 位整型	返回 1 则指令发送成功， 返回-1 则指令发送异常。
说明：注意使能马达。			

相关串口通讯指令如下所示

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型	/
发送	0x82	高位			低位	绝对定位	WO
接收	0x82	高位			低位	确认收到绝对定位	/
	0x50	0	0	0	1	超过软限位最小坐标	
		0	0	0	2	超过软限位最大坐标	

备注：

（1）旧版本程序，接收的数据类型为 0x81，表示反馈当前位置。

13.1.2 相对定位

以坐标当前位置为参考移动设定的距离。

相关 DLL 函数如下所示：

函数名	DLL_PositionRelativeMove		
函数原型	int DLL_PositionRelativeMove (unsigned int Address, int Position)		
函数功能	相对移动（以当前平台当前位置为参考进行移动）		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Position	有符号 32 位整型	位移距离设置值，正负则决定位移的方向，单位为 counts， 位置反馈的分辨率
返回值		有符号 32 位整型	返回 1 则指令发送成功， 返回-1 则指令发送异常
说明：注意使能马达			

相关串口通讯格式如下所示

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型， ±代表方向	/
发送	0x83	高位			低位	相对定位	WO
接收	0x83	高位			低位	确认收到相对定位	/
	0x50	0	0	0	3	超过软限位最小坐标	
		0	0	0	4	超过软限位最大坐标	

备注：

（1）旧软件版本，接收的数据类型为 0x81，表示反馈当前位置。

13.2 连续运动（JOG）

连续运动是指：发送 JOG 指令到驱控器，驱控器则一直驱动位移台移动，直至收到停止指令。另外收到停止指令后，会根据当前的速度逐渐降速，直至达到预设速度速度后才停止。所以会有一定的延迟性，因为当高速时，强制停止，会引起位移台震动。

相关 DLL 相关函数如下所示：

函数名	DLL_JOG_Move		
函数原型	int DLL_JOG_Move (unsigned int Address, unsigned char CMD)		
函数功能	JOG 模式移动		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	CMD	字符型	“L”左方向位移， “R”右方向位移， “T”停止位移
返回值		有符号 32 位整型	返回 1 则指令发送成功， 返回-1 则指令发送异常

相关串口通讯格式如下所示

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	/	/
发送	0x8E	0	0	0	0x4C (L)	“-”向 JOG 移动	WO
					0x52 (R)	“+”向 JOG 移动	
					0x53 (S)	停止 JOG 移动	
		高位			低位	数据区为未定义模式，都为停止 JOG 移动	
接收	0x8E	0	0	0	0x4C (L)	确认收到 JOG 模式移动	/
					0x52 (R)	确认收到 JOG 模式移动	
					0x53 (S)	确认收到停止 JOG 模式移动	
		高位			低位	确认收到停止 JOG 模式移动	

备注：

（1） 旧软件版本，接收的数据类型为 0x81，表示反馈当前位置。

13.3 停止运动

在运动过程中，可将位移台停止运行，会根据当前的速度逐渐降速，直至达到预设速度速度后才停止。所以会有一定的延迟性，因为当高速时，强制停止，会引起位移台震动。

相关串口通讯格式如下所示

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 位整型	/
发送	0x7C	0	0	0	0x53 (‘S’)	停止运动	WO
接收	0x7C	0	0	0	低位	停止运动	/
	0x50	0	0	0	36	输入错误	

13.4 回原点运动

- 寻找开机标记点。因为编码器信号是增量式，上电后没有参考点，可以进行寻找标记点，以该点设为上电时的参考点。具体查看上电起始找零点说明。
- 不同于回 0 定位。回 0 运动是指定位到坐标 0（绝对定位）。

13.5 使能/失能

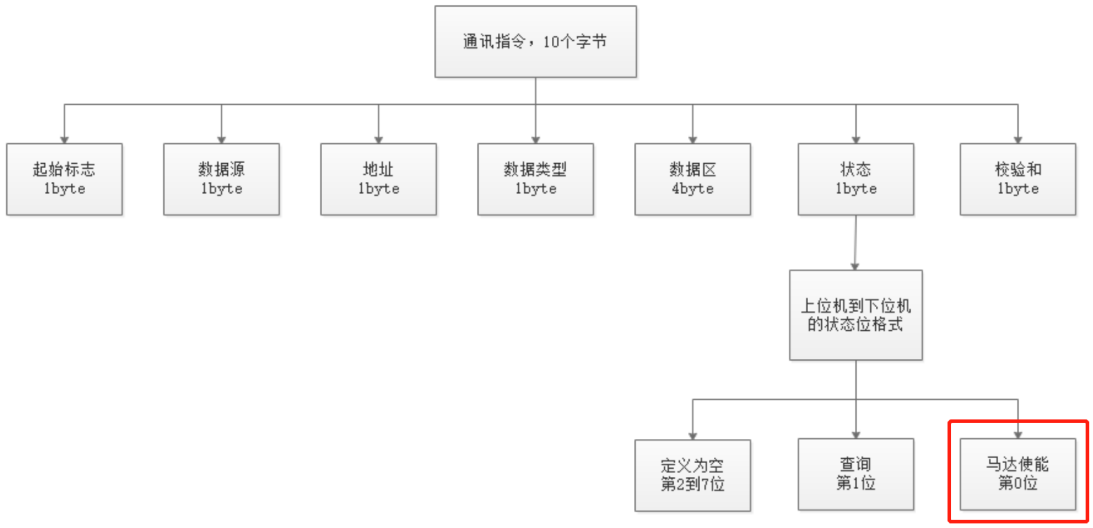
- 可通过设置控制马达使能（上电）或失能（下电）。
- 马达使能后默认进入闭环控制，当位置偏移后会进行自动校准（滑动位移台台面可能会引起警报）。另外可配置为定位完成后不启动碰撞检测，例如：载物台换样片。

相关 DLL 函数如下所示：

函数名	DLL_AxisEnable		
函数原型	int DLL_AxisEnable (unsigned int Address, unsigned char KG)		
函数功能	控制器使能与失能设置		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	KG	字符型	输入“K”设置马达使能， 输入“G”设置马达失能
返回值		有符号 32 位整型	设置成功返回 1，否则返回-1
说明：设置失败需查询是否已连接上驱控器。			

串口通讯中控制使能方式与状态字节(通讯包中第 8 个字节相关)中第 0 位相关。

图 13.2 使能位在通讯指令中位置



13.6 I/O 口功能

- I/O 接口为光耦结构。
- 可用于接入外部的限位光耦进行限位。
- 可用于紧急停止。
- 用于进入强制更新程序模式；查看程序更新章节。
- 输出脉冲（扫描运动，触发工业相机）；查看扫描模式（脉冲输出）章节。
- 脉冲控制（一路为脉冲输入，一路为方向输入）；查看脉冲控制模式章节。

备注：

（1）V2.3、V3.1、V3.5 版本驱控器，脉冲输出与脉冲控制只能选其一（与驱控器上连接有关）；

（2）JC-C03-P01 版本支持同时脉冲输出与脉冲控制。

13.7 软件限位

通过软件限位，限制位移台的行程。（物理上位移台的行程不变，但通过软件操作时，无法移动到限位的行程）。

比如位移台的实际行程为 1mm（50nm 光栅），1mm/0.00005mm=20000，全行程为 20000counts，取位移台中心为 0 点，则坐标点为-10000 到+10000。

设置最大坐标点为 7500，最小坐标点为-7500，通过上位机驱动时，最大行程为 0.75mm。

说明：

最大/最小坐标点生效需要有两个前提：

（1） 找完 0 点

（2） 最大坐标点≠最小坐标点（设置最大/最小坐标点都为 0 时，关闭软件限位）

相关 DLL 函数如下所示

函数名	DLL_SetBoundary		
函数原型	int DLL_SetBoundary(unsigned int Address, int Value, unsigned int Mode)		
函数功能	设置最大/最小边界点		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Value	有符号 32 位整型	最大/最小边界点的坐标值
	Mode	无符号 32 位整型	Mode 为 1 设置最小边界点， Mode 为 2 设置最大边界点。
返回值		有符号 32 位整型	返回 1 则设置成功， 返回 0 则没有收到返回， 返回-1 则设置失败。

函数名	DLL_GetBoundary		
函数原型	int DLL_GetBoundary (unsigned int Address, unsigned int Mode)		
函数功能	查询最大/最小边界点		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Mode	无符号 32 位整型	Mode 为 1 时查询最小边界点， Mode 为 2 时查询最大边界点。
返回值		有符号 32 位整型	查询成功时返回最小/最大边界点， 查询失败时返回-1。

相关串口通讯格式如下所示

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 位整型	/
发送	0x87	高位			低位	最小边界点	RW
接收	0x87	高位			低位	最小边界点	/
	0x50	0	0	0	20	错误，最小边界点大于 最大边界点	

含义	数据 类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 位整型	/
发送	0x88	高位			低位	最大边界点	RW
接收	0x88	高位			低位	最大边界点	/
	0x50	0	0	0	21	错误，最大边界点小于 最小边界点	

13.8 碰撞检测

驱控器通过一定时间内位置坐标变化、驱动功率、速度、加速度等因素进行检测位移台是否发生碰撞异常。比如使能下，用手滑动、按住台面，碰撞到硬限位等时，驱控器将抛出异常，此时红灯。碰撞检测的灵敏度通过“碰撞系数”调节。

相关的 DLL 函数有：
《DLL_SetBaseValue》，形参 mode 为 4，用于设置“开启/关闭碰撞检测”。
《DLL_GetBaseValue》，形参 mode 为 4，用于查询“碰撞检测状态”。
《DLL_SetBaseValue》，形参 mode 为 9，用于设置“碰撞系数”。
《DLL_GetBaseValue》，形参 mode 为 9，用于查询“碰撞系数”。

相关串口通讯格式如下所示

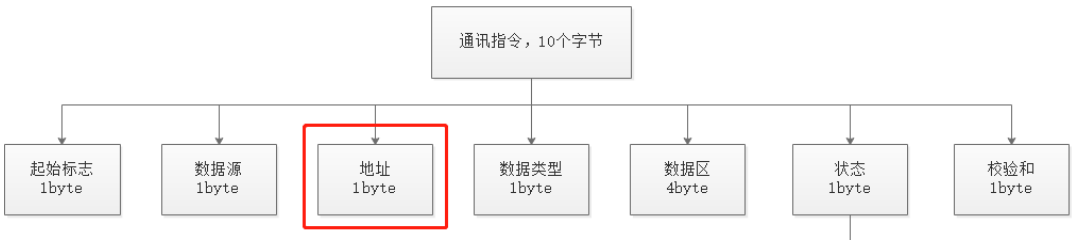
含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 位整型	/
发送	0x57	0	0	0	0	设置关闭碰撞检测	RW
					7	设置打开碰撞检测	
接收	0x57	0	0	0	0	碰撞检测打开	/
					7	碰撞检测关闭	
	0x50	0	0	0	37	输入错误	

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 位整型	/
发送	0x59	高位			低位	设置碰撞系数	RW
接收	0x59	高位			低位	碰撞系数	/

13.9 多轴运用

多轴运用时通过 485 接口，将通讯线串接起来。此时需将每个驱控器设置单独的轴号（地址）。

图 13.3 通讯包中地址字节



相关 DLL 函数如下所示：

函数名	DLL_ChangeAxisNum		
函数原型	int DLL_ChangeAxisNum(unsigned int Address,unsigned int NewAdd)		
函数功能	设置驱控器轴号		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器轴号，范围为[1,8]
	NewAdd	无符号 32 位整型	设置的新轴号，范围为[1,8]
返回值		有符号 32 位整型	返回 1 则设置成功， 返回 0 则未收到校验， 返回-1 则设置失败。
注意：设置成功后，返回指令中轴号为新设置的轴号			

相关串口通讯格式如下所示

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 位整型	/
发送	0x8C	0	0	0	*	设置驱控器新轴号， 范围 1 到 9	RW
接收	0x8C	0	0	0	*	驱控器轴号	/
	0x50	0	0	0	39	错误，轴号范围不对	

13.10 速度设置

驱控器速度根据模式分为 4 种类型速度，分别为定位速度，JOG 速度，找零位速度，扫描速度，对应所处模式下的速度。相对运动跟绝对运动速度为定位速度。

相关 DLL 函数如下所示

函数名	DLL_SetSpeed4		
函数原型	int DLL_SetSpeed4(unsigned int Address, unsigned char Type, unsigned int countsPerS)		
函数功能	速度设置（最低分辨率为 10000count/s，速度设置范围 30000~600000count/s），其他范围需咨询工程师		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Type	字符型	“V”设置定位速度， “J”设置 JOG 速度， “F”设置开机找零位速度， “S”设置扫描速度。
	countsPerS	无符号 32 位整型	速度设置值
返回值		有符号 32 位整型	返回 1 设置成功， 返回-1 设置失败。

相关串口通讯格式如下所示

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0x7A	高位			低位	设置定位速度 单位 counts/s 范围 1～1000000	RW
接收	0x7A	高位			低位	定位速度	/
	0x50	0	0	0	10	设置的定位速度过大	
		0	0	0	11	设置的定位速度过小	

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0x9D	高位	0	0	低位	设置 JOG 速度 输入范围 2～100 单位（×10）counts/ms	RW
接收	0x9D	高位			低位	JOG 速度	/
	0x50	0	0	0	48	设置的 JOG 速度过大	
		0	0	0	49	设置的 JOG 速度过小	

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0x7B	高位			低位	设置 JOG 速度 单位 counts/s 范围 1～1000000	RW
接收	0x7B	高位			低位	JOG 速度	/
	0x50	0	0	0	48	设置的 JOG 速度过大	
		0	0	0	49	设置的 JOG 速度过小	

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0x9B	高位			低位	设置找零速度 输入范围 2～100 单位（×10）counts/ms	RW
接收	0x9B	高位			低位	找零速度	/
	0x50	0	0	0	18	设置的找零速度过大	
		0	0	0	19	设置的找零速度过小	

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0x79	高位			低位	设置找零速度 单位 counts/s 范围 20000～1000000	RW
接收	0x79	高位			低位	找零速度	/
	0x50	0	0	0	18	设置的找零速度过大	
		0	0	0	19	设置的找零速度过小	

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0x9E	高位			低位	设置扫描速度 输入范围 2～100 单位（×10）counts/ms	RW
接收	0x9E	高位			低位	扫描速度	/
	0x50	0	0	0	33	设置的扫描速度过大	
		0	0	0	34	设置的扫描速度过小	

13.11 旋转台使用

旋转台使用，行程 360°的旋转台使用，到达一定距离会重置为 0。因为旋转台从 0°移动到 360°时，实际上旋转台回到 0°的位置点，如果不置 0，旋转的角度随之递增（超过 360°）。

相关串口通讯格式如下所示

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0x58	高位			低位	设置重置点	RW
接收	0x58	高位			低位	重置点	/
	0x50	0	0	0	94	未进入开发者模式	
		0	0	0	43	输入范围错误	

13.12 位置坐标

驱控器闭环使用时，需接入增量式的位置反馈。可将位置坐标传递到上位机。另外可设置将位置坐标取反，比如 1000 修改为-1000。

相关 DLL 函数如下所示

函数名	DLL_ReadPosition		
函数原型	int DLL_ReadPosition(unsigned int Address)		
函数功能	读取指定轴的位置		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
返回值		有符号 32 位整型	返回位置信息

相关串口通讯格式如下所示

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型	/
发送	0x81	0	0	0x54	0x50	查询位置坐标	RO
接收	0x81	高位			低位	当前位置坐标	/

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型	/
发送	0x60	0	0	0	0	设置位置坐标相位为正	RW
	0x60	0	0	0	1	设置位置坐标相位为反	
接收	0x60	高位			低位	位置坐标相位	/

十四、错误（异常）反馈机制说明

异常分为运行异常和参数设置类异常。

14.1 运行异常

运行异常时，下位机反馈到上位机的状态字节中异常位为 1（状态字节中第 6 位），此时可通过查询运行状态指令查询异常，反馈的数据类型为 0x54，数据区 4 个字节构成为 32bit，每个 bit 代表一种运行异常。异常时进行软件清除异常。

表 12.1 状态码异常含义

31	30	29	28
读取数据异常	重启中	通信异常	保留
27	26	25	24
保留			步进异常
23	22	21	20
保留			
19	18	17	16
保留	高温异常	过流异常	满载过长异常
15	14	13	12
PWM 异常	运动模式异常	外部紧急停止	找 0 异常
11	10	9	8
开环占空比异常	开环频率异常	保留	
7	6	5	4
保留			
3	2	1	0
编码器异常	驱动方向异常	触发+边界	触发-边界

相关 DLL 函数如下所示

函数名	DLL_GetStatusError		
函数原型	unsigned int DLL_GetStatusError (unsigned int Address, unsigned char* Output)		
函数功能	查询驱控器的运动状态		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	查询驱控器的轴号（1,2,3...）
	Output	字符指针	保存状态异常类型的说明
返回值		无符号 32 位整型	返回 0 则状态正常， 返回其他正整数为异常状态码， 返回-1 则读取失败。

函数名	DLL_ClearMotorError		
函数原型	int DLL_ClearMotorError(unsigned int Address)		
函数功能	查清除驱控器的异常状态		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	查询驱控器的轴号（1,2,3...）
返回值		有符号 32 位整型	返回 1 则设置成功， 返回-1 则设置失败。
说明：当驱动异常触发时，驱控器会标记异常，此时该轴的驱控器会受到保护，无法响应运动控制指令。此函数会清除该轴的异常标记，使该函数能再次响应运动控制指令。但是未解决异常原因的情况下，再次驱动往往会再出现异常，可通过 DLL_GetStatusError 函数找出异常原因。			

相关串口通讯格式如下所示

含义	数据类型	数据区				数据区含义	读写属性 RO：仅读 WO：仅写 RW：可读可写
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型，	/
发送	0x54	0	0	0	0	查询/设置异常码	RW
接收	0x54	高位			低位	返回异常码	/

14.2 参数设置异常

当输入参数错误时，下位机反馈上位机异常错误信息（数据类型 0x50，表 14.2）。

表 14.2　异常反馈表

数据类型	数据区内容				错误信息
	Byte4	Byte5	Byte6	Byte7	
0x50	0	0	0	1	绝对坐标运动，超出最小边界点
				2	绝对坐标运动，超出最大边界点
				3	相对坐标运动，超出最小边界点
				4	相对坐标运动，超出最大边界点
				10	定位速度设置过大
				11	定位速度设置过小

				12	速度 2 设置过大
				13	速度 2 设置过小
				16	起始 PWM 设置过大
				17	起始 PWM 设置过小
				18	找标记点速度过大
				19	找标记点速度过小
				20	最小边界点不能大于当前最大边界点
				21	最大边界点不能小于当前最小边界点
				23	加速度过大
				24	加速度过小
				25	P 范围设置有误
				26	I 范围设置有误
				27	D 范围设置有误
				28	细调模式 PWM 设置错误
				29	细调模式范围过小
				30	细调模式脉宽设置错误
				33	扫描速度过大
				34	扫描速度过小
				35	调速周期设置范围有误
				36	Stop 输入错误
				37	边界开关设置错误
				38	细调模式设置 BA 输入错误
				39	轴号设置错误
				40	滤波数输入错误
				41	设置标记点输入参数错误
				42	eeprom 操作输入参数错误

				43	重置点输入错误
				44	查询软件/硬件版本号输入错误
				45	惯性系数设置错误
				46	调节间隙过长
				47	调节间隙过小
				48	JOG 速度设置过大
				49	JOG 速度设置过小
				56	碰撞找零误差范围设置过大
				57	碰撞找零误差范围设置过小
				60	开环驱动频率过大
				61	开环驱动频率过小
				62	开环驱动脉宽过大
				63	开环驱动脉宽过小
				64	开环驱动幅值过大
				70	脉冲输出间隔过大
				71	脉冲输出间隔过小
				72	脉冲输出脉宽过大
				73	脉冲输出脉宽过小
				74	脉冲输出掩盖数量过大
				75	脉冲输出掩盖数量过小
				76	扫描起始点超过最小边界点
				77	扫描起始点超过最大边界点
				80	延迟找零时间设置过大
				92	存储操作输入错误
				94	未进入开发者模式
				95	使能提示

				96	读取 IC 系列码失败
				97	驱动频率设置错误
				98	失能提示
				99	重启失败
				100	保存失败
				153	波特率设置过小
				154	波特率设置过大

十五、速度曲线说明

图 15.1 时间/位移图

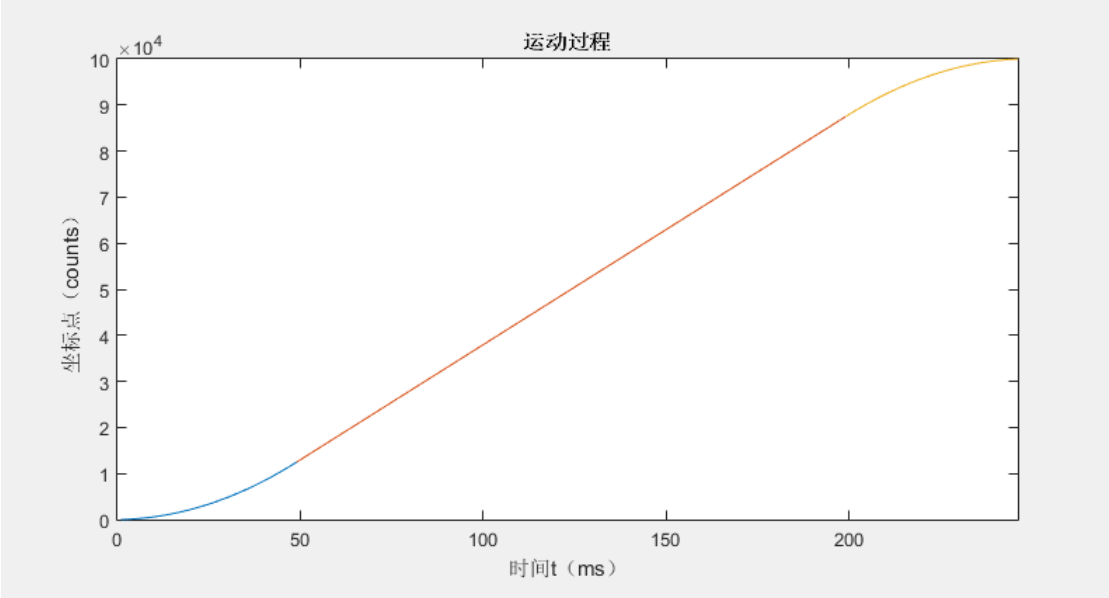


图 15.2 时间/速度图

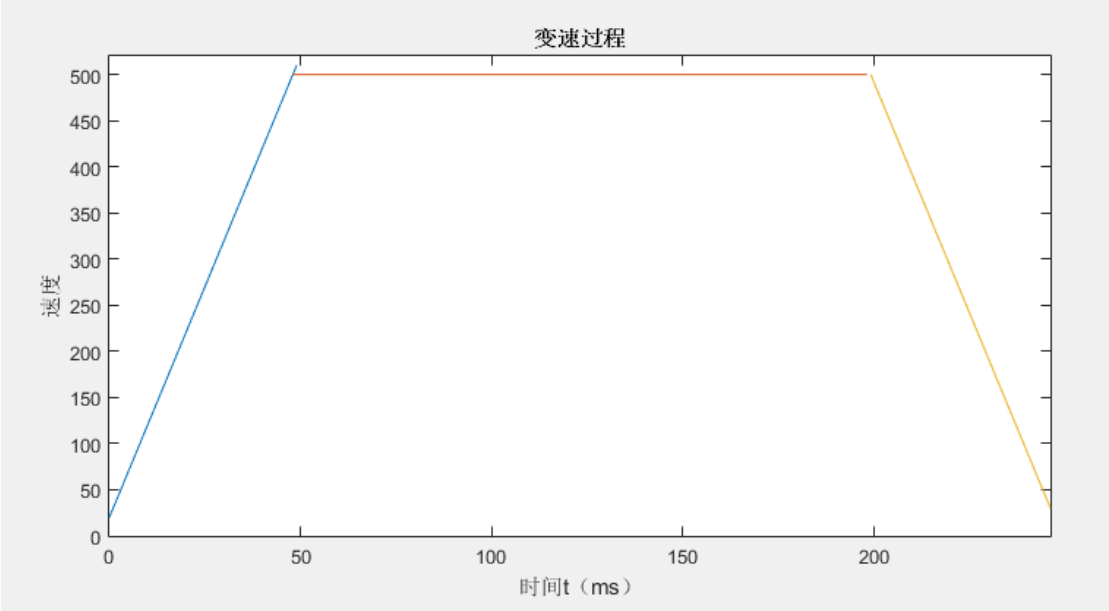


图 12.2

定位过程中，驱控器根据最大速度、最低速度与加速度进行仿真运动曲线，运动过程中根据预设速度进行调节。

当加速度一定时，预设速度将从最低升至最高；如果位移量过低，不足以提升到最高速度，则将加减速过程根据路程折中计算。

例子 1：图 15.1 和图 15.2 中，位移量 S=100000(counts)，最低速度 Vmin=20(counts/ms)，最高速度 Vmax=300(counts/ms)，加速度 a=10(counts/ms^2)；假设 1counts=50nm，则

S=100000*0.00005mm=5mm；

Vmax=300*0.05(um/ms)=15(um/ms)=15(mm/s)；

Vmin=20*0.05(um/ms)=1(um/ms)=1(mm/s)；

图 15.3 时间/位移图

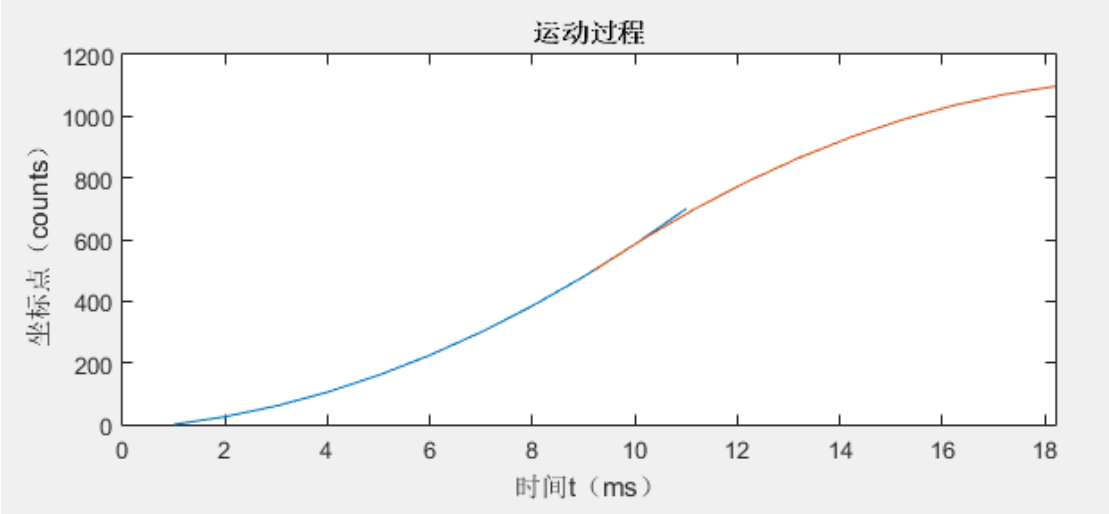
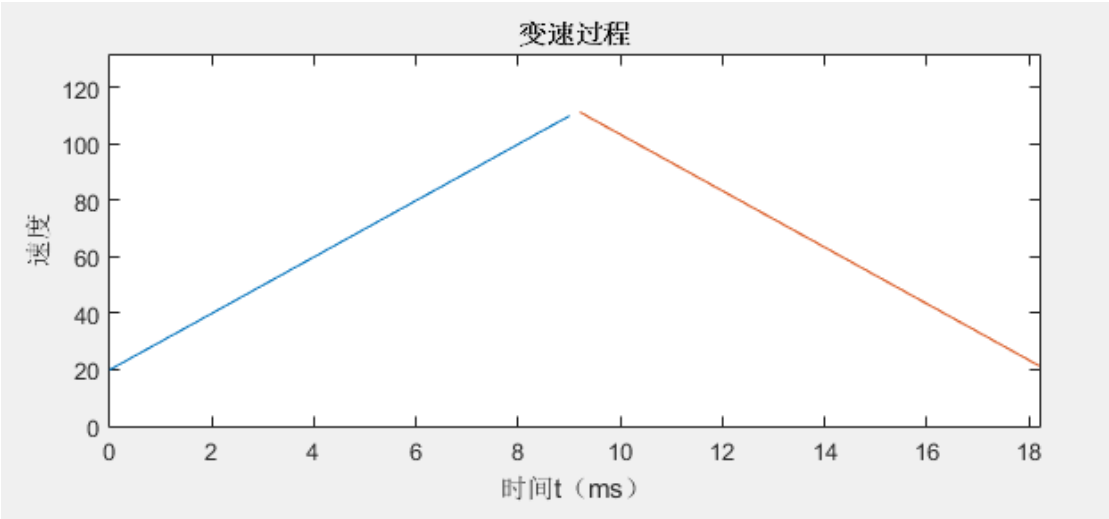


图 15.4 时间/速度图



例子 2：图 15.3 和图 15.4 中，位移量 S=1200(counts)，最低速度 Vmin=20(counts/ms)，最高速度 Vmax=300(counts/ms)，加速度 a=10(counts/ms^2)；
加速到最高速所需的时长 t=(Vmax-Vmin)/a=(300-20)/10=28ms;
加速到最高速度所需的位移 L=Vmin*t+a*(t^2)/2=20*28+10*(10^2)/2=1060;
因为加速过程>路程的一半（ L>(S/2) ），则将全程分为两个部分（加速和减速过程）。则最大速度为Vmax=sqrt（（Vmin^2）+2*a*S/2）;

十六、参数存储说明

存储参数注意点：

1、存储指令有滞后性（存储完成后才反馈）

2、存储过程消耗时间（大于 40ms 小于 80ms）；

3、建议存储时，到下一次通讯间隔时间>=100ms 或者接收到驱控器反馈后再进行下一次通信,否则会导致 485 通讯（半双工通讯）的收发冲突（485 通讯失效，需要断电重启才能再次使用）。

相关 DLL 函数如下所示

函数名	DLL_SetEPW		
函数原型	int DLL_SetEPW(unsigned int Address,unsigned char DS)		
函数功能	保存驱控器的参数		
	变量名	变量类型	说明
形参	Address	有符号 32 位整型	串口号（1,2,3...）
	DS	字符型	固定输入 “S”
返回值		有符号 32 位整型	返回 1 则设置成功， 返回 0 则未收到校验， 返回-1 则设置失败。

相关串口通讯格式如下所示

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型，	/
发送	0x84	0	0	0	0x57 (‘W’)	存储数据	WO
					0x41 (‘A’)	存储备份数据	
					0x48 (‘H’)	恢复出厂数据	
					0x44 (‘D’)	删除存储数据	
接收	0x84	0			0	存储完成	/
	0x50	0	0	0	100	保存失败	
					94	未进入开发者模式， 保存失败	
					92	输入参数错误	
					98	需失能马达才能操作	

十七、上电起始找零点说明

17.1 开机找零点说明

因位置反馈为增量式，每次上电需要将光栅尺上的标记点作为起始标志点，进而设置上电零位。

相关 DLL 函数如下所示

函数名	DLL_SetFIMode_S		
函数原型	int DLL_SetFIMode_S(unsigned int Address,unsigned char Mode)		
函数功能	设置开机寻零		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Mode	无符号 8 位整型	'Y'启动开机找零 'F'启动开机反向找零 'N'关闭开机找零 'S'单次启动 '+'单次正向启动 '-'单次反向启动
返回值		有符号 32 位整型	设置成功返回 1， 没有收到返回 0， 参数错误返回-1。

函数名	DLL_GetFIMode_S		
函数原型	int DLL_GetFIMode_S(unsigned int Address)		
函数功能	查询指定轴的开机启动模式		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
返回值		有符号 32 位整型	返回 1 则是启动开机正向找零位， 返回 2 则是启动开机反向找零位， 返回 3 则是关闭开机找零位， 返回 0 则是未收到校验数据， 返回-1 则是设置参数错误。

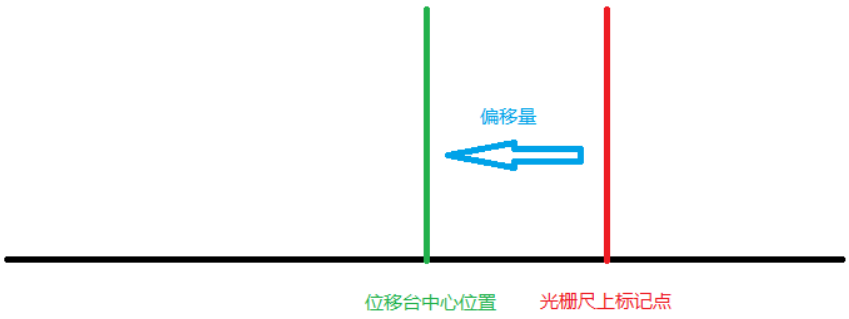
相关串口通讯格式如下所示

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 位整型	/
发送	0x8B	0	0	0x59 ('Y')	0x59 ('Y')	设置开机找零点(反向)	RW
				0	0x59 ('Y')	设置开机找零(正向)	
					0x4E ('N')	取消找领	
					0X53 ('S')	启动找零(单次使用)	
					0x2B ('-')	启动找零(反向，单次使用)	
					0x2D ('+')	启动找零(正向，单次使用)	
接收	0x8B	高位			低位	确认收到设置开机找 0	/
	0x50	0	0	0	41	形参输入错误	

17.2 开机零点偏移指令说明

因为光栅尺上的原点（标记点）相对平台中心有一定的偏差（与安装相关），所以需要设置一定偏移，将位移台开机 0 点处于中心。

图 17.1 偏移量说明图



相关 DLL 函数如下所示：
查看《DLL_SetBaseValue》，形参 Mode 的值为 1；
查看《DLL_SetStepValue》，形参 Mode 的值为 1；

相关串口通讯格式如下所示：

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 位整型 ±表示偏移方向	/
发送	0x92	高位			低位	开机零点偏移量	WO
接收	0x92	高位			低位	确认收到开机零点偏移量	/

另外可通过指令可设置当前位置为下一次寻零后的零点（前提为上一次有寻零完成，则将自动设置偏移量）

相关串口通讯格式如下所示：

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 位整型	/
发送	0x7D	0	0	0	0x48 ('H')	设置当前位置为上电零点	WO
接收	0x7D	0	0	0	0	确认收到当前位置为上电零点	/

备注：
（1）V3.7.00 版本以前，反馈信息为位置信息；
（2）软件版本 V3.6.17 及以后支持。

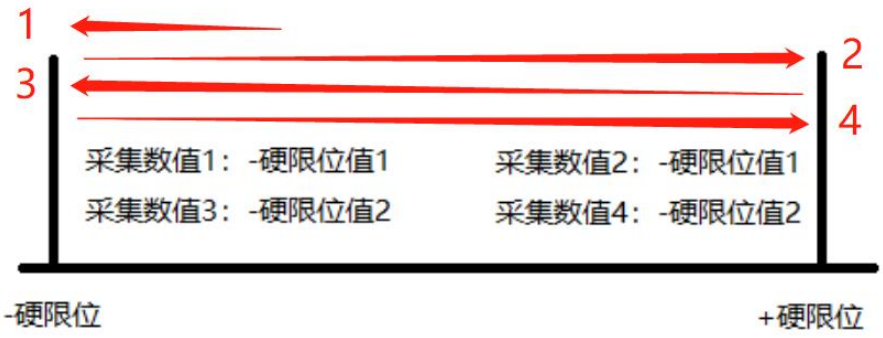
17.3 关于找完在原点之后，坐标点不为 0 原因

（1） 完成开机找 0 后，由于位移台存在惯性，会滑动一定的距离（当马达没有使能时）；
（2） 避免办法：找标记点时，使能马达（闭环控制）；
（3） 开机找 0 过程中，如果需要中止，参考“取消找 0”指令或者拔出马达插头。

17.4 碰撞找中心点说明

通过分别碰撞两端至少两次，当在±两端检测到极限点误差范围在一定之内，则判断检测正常，进而计算出位移台中心点。如果有加上软件限位，则会进行对行程的判断。

图 17.2 碰撞找中心点示意图



备注：默认碰撞超过 10 次（5 个循环）无法找中心点完成，则抛出异常停止运行。

相关串口通讯格式如下所示：

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 位整型	/
发送	0x5D	0	0	0	*	开机找零碰撞上限次数， 输入范围 0～100	RW
接收	0x5D	0	0	0	*	确认收到找零碰撞上限次数	/

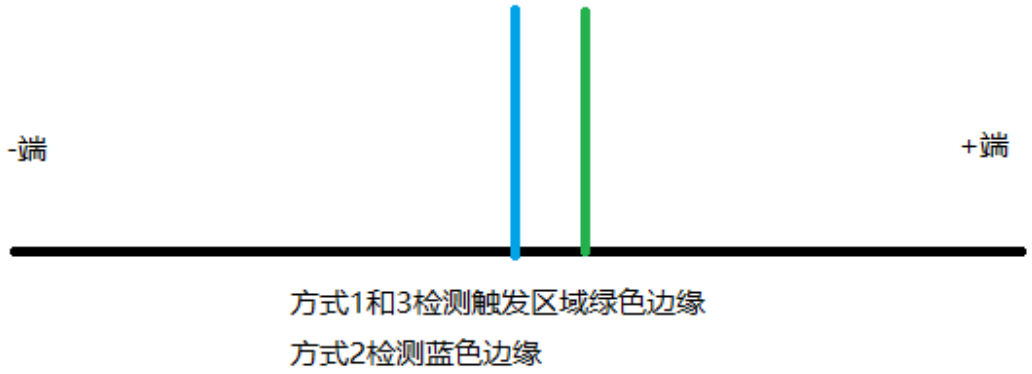
备注：

（1） 输入值超过 255，强制为 255。

17.5 驱控器开机找原点方式

（1） 方式 1 为检测触发区域 + 端
（2） 方式 2 为检测触发区域 - 端
（3） 方式 3 为检测触发区域 + 端（检测边界范围）
（4） 方式 4 为通过碰撞±两端，计算中心点

图 17.3 寻零方式说明图



备注：
（1）修改此项需联系工程师。

17.6 设置开机延迟找零

可设置上电延迟一段时间后再进行找零。

对应 DLL 函数为：DLL_SetBaseValue，mode 值为 3。

相关串口通讯格式如下所示

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 位整型	/
发送	0x6E	高位			低位	开机延迟找零时间 单位 ms 输入范围 0～50000	RW
接收	0x6E	高位			低位	开机延迟找零时间	/
	0x50	0	0	0	80	设置时间过大	

十八、扫描触发模式（脉冲输出）

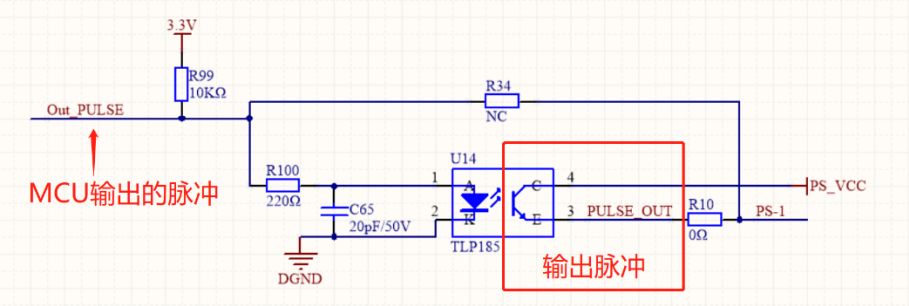
18.1 扫描触发说明

脉冲输出功能运用于输出脉冲触发工业相机、触发激光器等运用。

18.2 扫描触发硬件原理

驱控器的脉冲输出采用光耦隔离，需注意连接。

图 18.1　驱控器脉冲输出电路



18.3 扫描触发硬件连接

图 18.2　驱控器 I/O 接口

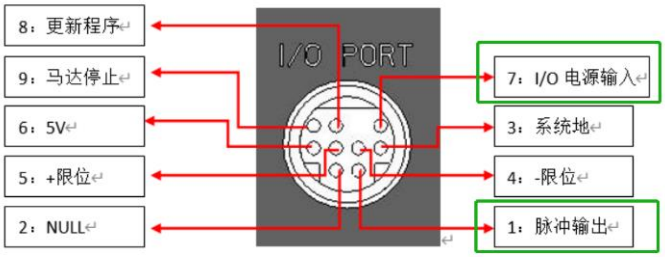
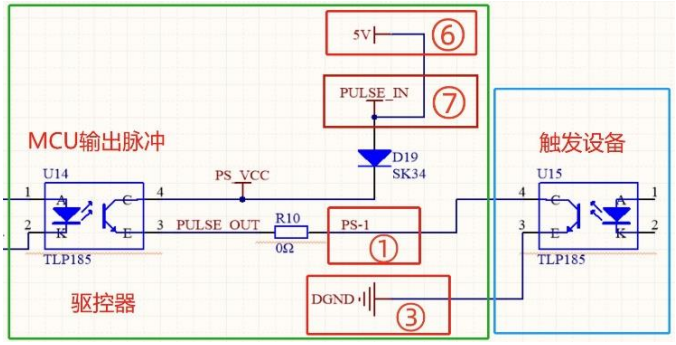


图 18.3 连接原理图



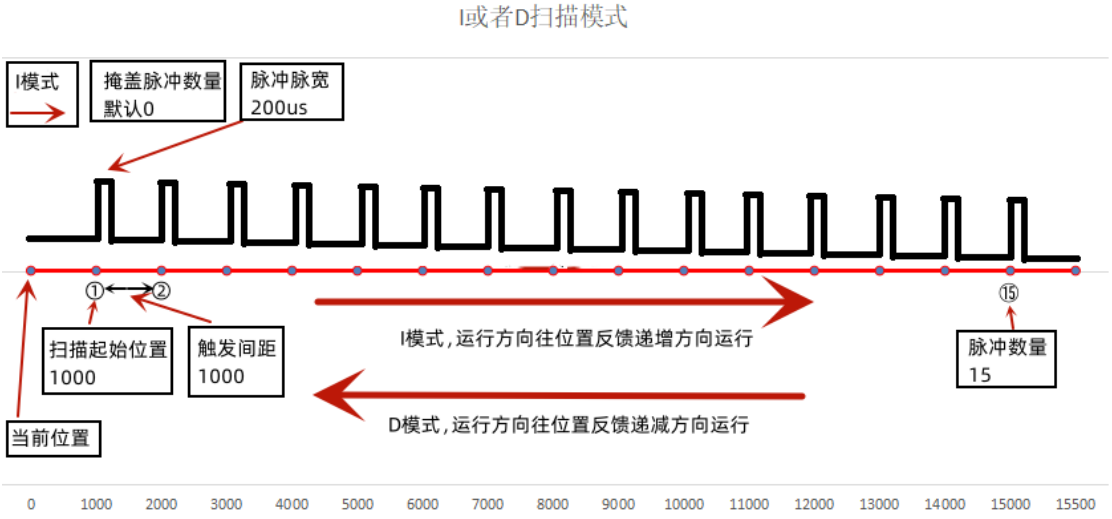
- 触发设备的脉冲输入一般是光耦隔离。
- 一般使用时，将驱控器 I/O 接口上⑥跟⑦短接，①接触发设备的触发信号输入引脚，③接触发设备的触发信号输出引脚。
- 当使用＞5V 系统时，需注意添加限流电阻。

18.4 扫描触发模式说明

18.4.1 扫描触发模式 I 模式与 D 模式说明

- I 模式跟 D 模式运用于单轴脉冲输出。
- I 跟 D 控制扫描的方向。I 为位置反馈递增方向移动，D 为位置反馈递减方向移动。
- “扫描起始点”为第一个脉冲输出点的位置。不一定需要移动到扫描起始点在进行启动扫描触发模式，驱控器会自动将位移台先移动到“扫描起始点”，再按照 I 或者 D 模式进行相应移动。
- “脉冲脉宽”为输出脉冲的脉宽。
- “脉冲数量”决定扫描触发的脉冲数量。
- “触发间距”为相隔一定距离输出脉冲的距离。
- “脉冲数量”跟“脉冲间距”决定扫描长度。
- 最后停止的位置跟“扫描速度”有关，当所需的脉冲输出完毕后，位移台将进行减速运动，直至停止。

图 18.4 D 模式示例



18.4.2 触发模式 P 模式与 N 模式说明

- P 模式跟 N 模式运用于蛇形扫描（双轴）脉冲输出。
- P 跟 N 控制扫描触发移动的方向。P 为位置反馈递增方向移动，N 为位置反馈递减方向移动。
- 使用串口指令调用顺序：设置参数（扫描起始位置、触发间距等）→启动 P 模式→启动 N 模式→启动 P 模式→启动 N 模式・・・，以此类推。

图 18.5　P 模式示例（X 轴为触发轴）

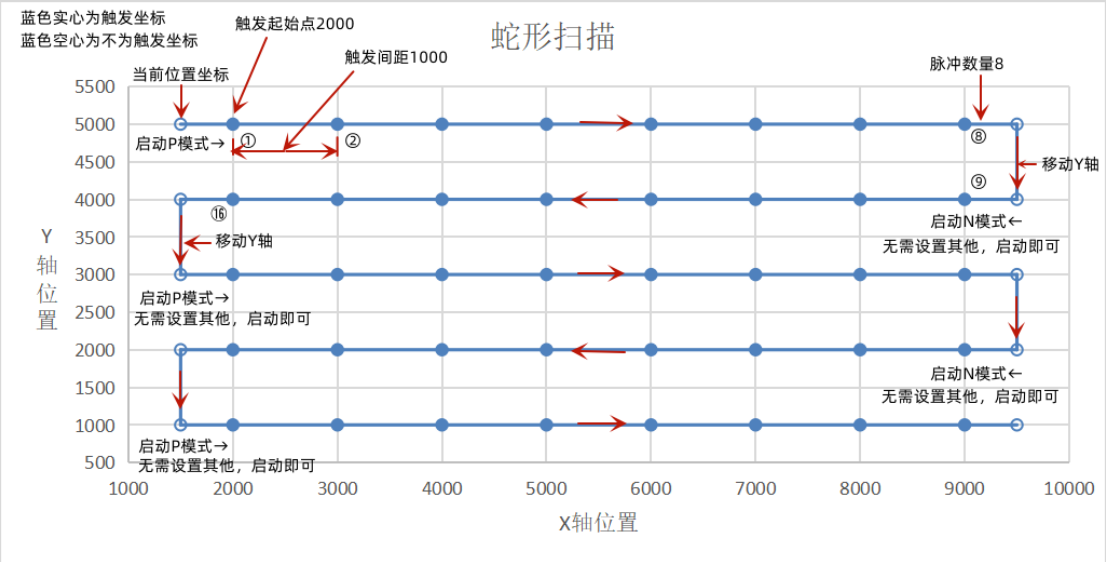
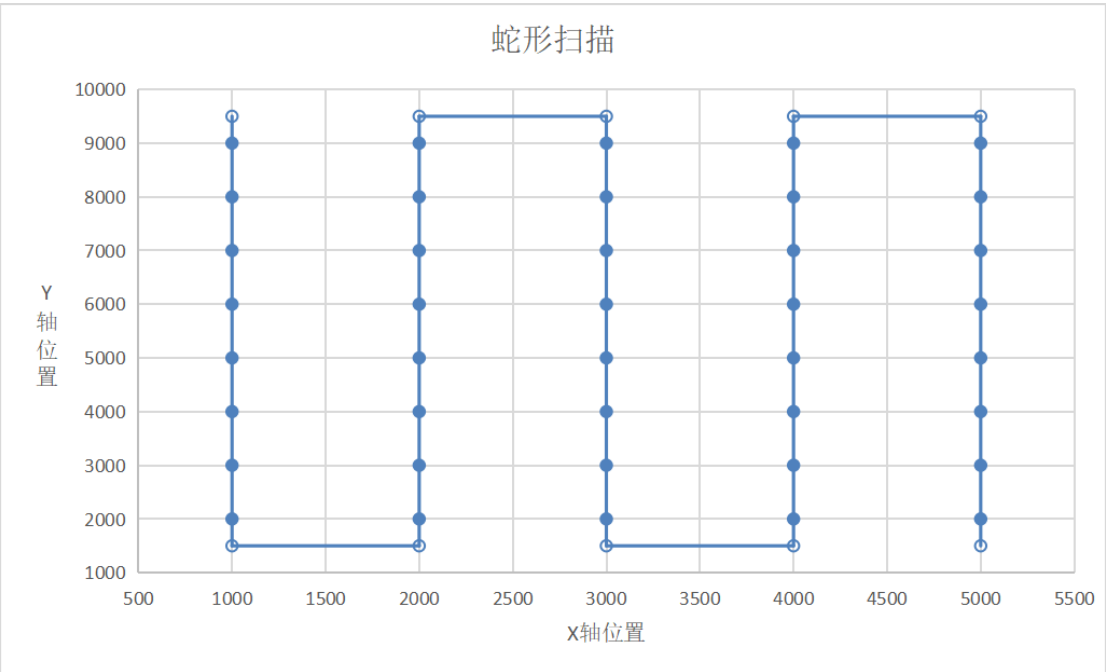


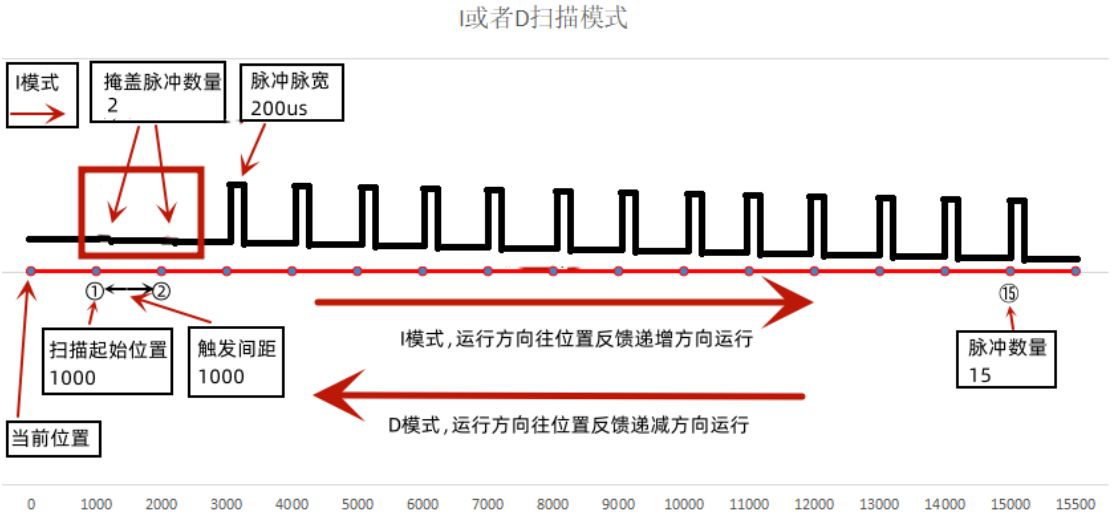
图 18.6　P 模式示例（Y 轴为触发轴）



18.4.3 掩盖脉冲使用说明

因某些触发设备需要匀速触发，但位移台可能存在加速过程（但当前位置移动到扫描起始点跟后续移动的方向不一致时，相当于位移台开始触发前几个脉冲可能会位于加速区间），这时可以对前几个触发脉冲进行屏蔽。

图 16.7　D 模式掩盖使用示例



18.5 到位脉冲输出

脉冲输出还可以设置为定位完成时发出脉冲。

18.6 脉冲输出通讯接口说明

相关 DLL 函数如下所示：

函数名	DLL_Mod2ScanControl2		
函数原型	int DLL_Mod2ScanControl2(unsigned int Address, unsigned char CMD, unsigned char control)		
函数功能	控制单轴扫描脉冲触发驱动		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	CMD	字符型	“I”为正向扫描， “D”为反向扫描。
	control	无符号 8 位整型	默认为 0
返回值		有符号 32 位整型	设置成功返回 1， 设置失败返回 0。
说明：该函数需要的参数为扫描轴的轴号和扫描的方向，而扫描还需要更多的参数，所以在使用此函数扫描之前，请先调用其他函数设置完扫描功能所需的其他参数。			

函数名	DLL_Mod2ScanStartPosition		
函数原型	int DLL_Mod2ScanStartPosition (unsigned int Address, int Position)		
函数功能	设置单轴扫描的起始点		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Position	有符号 32 位整型	扫描起始点的坐标
返回值		有符号 32 位整型	返回 1 设置成功， 返回 0 设置失败。

函数名	DLL_Mod2ScanStartPulseSet		
函数原型	int DLL_Mod2ScanStartPulseSet(unsigned int Address, unsigned int pitch, unsigned int pitch_num)		
函数功能	设置单轴扫描的脉冲触发间距和脉冲触发个数		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	pitch	无符号 32 位整型	脉冲触发间距
	Pitch_num	无符号 32 位整型	脉冲数量
返回值		有符号 32 位整型	返回 1 设置成功， 返回 0 设置失败。

函数名	DLL_Mod2ScanPulseWidthSet		
函数原型	int DLL_Mod2ScanPulseWidthSet (unsigned int Address, unsigned int Width)		
函数功能	设置单轴扫描的脉宽		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	轴号设置（1,2,3...）
	Width	无符号 32 位整型	脉宽,单位 us。
返回值		有符号 32 位整型	返回 1 设置成功， 返回 0 设置失败。

函数名	DLL_Mod2ScanMaskPulseSet		
函数原型	int DLL_Mod2ScanMaskPulseSet (unsigned int Address, unsigned int MaskPulse)		
函数功能	设置单轴扫描脉冲掩盖的数量		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	轴号设置（1,2,3...）
	MaskPulse	无符号 32 位整型	掩盖的数量
返回值		有符号 32 位整型	返回 1 设置成功， 返回 0 设置失败。

函数名	DLL_SetSpeed4		
函数原型	int DLL_SetSpeed4(unsigned int Address, unsigned char Type, unsigned int countsPerS)，其他范围需咨询工程师		
函数功能	速度设置（最低分辨率为 10000count/s，速度设置范围 30000~600000count/s）		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Type	字符型	设置的速度类型选择， “V”设置普通定位速度， “J”设置 JOG 位移速度， “F”设置开机找零位速度， “S”设置扫描速度。
	countsPerS	无符号 32 位整型	速度设置值
返回值		有符号 32 位整型	返回 1 设置成功， 返回-1 设置失败。

函数名	TAS_control		
函数原型	int TAS_control(int CMD)		
函数功能	蛇形扫描控制		
	变量名	变量类型	说明
形参	CMD	有符号 32 位整型	CMD 为 0 停止扫描； CMD 为 1 进行单次扫描； CMD 为 2 开始连续扫描； CMD 为 3 暂停扫描； CMD 为 4 恢复扫描； CMD 为 5 交换扫描轴与定位轴。
返回值		有符号 32 位整型	返回 1 指令发送成功， 返回 0 指令发送失败。

函数名	TAS_scanRangeSet		
函数原型	int TAS_scanRangeSet(int scanMin, int scanMax, int scanPulseInterval, int stepMin, int stepMax, int stepLenth, int stepPrecision)		
函数功能	设置蛇形扫描的参数		
	变量名	变量类型	说明
形参	scanMin	有符号 32 位整型	扫描轴最小位置点
	scanMax	有符号 32 位整型	扫描轴最大位置点
	scanPulseInterval	有符号 32 位整型	扫描轴脉冲间隔
	stepMin	有符号 32 位整型	定位轴最小位置点
	stepMax	有符号 32 位整型	定位轴最大位置点
	stepLenth	有符号 32 位整型	定位轴步长
	stepPrecision	有符号 32 位整型	定位轴精度

返回值		有符号 32 位整型	返回 1 设置成功， 返回-1 设置失败。

函数名	TAS_getStatus		
函数原型	int TAS_getStatus(int statusNum)		
函数功能	读取蛇形扫描状态信息		
	变量名	变量类型	说明
形参	statusNum	有符号 32 位整型	statusNum 为 1 查询是否处于扫描状态； statusNum 为 3 查询当前扫描行的行号； statusNum 为 4 查询定位轴轴号； statusNum 为 5 查询扫描轴轴号扫描
返回值		有符号 32 位整型	根据参数不同，返回值意义不同； 参数设置错误，返回-1； 参数设置 1，返回 0 则停止扫描，返回 1 则正在单次扫描，返回 2 则正在连续扫描，返回 3 则暂停扫描； 参数设置 3，返回当前扫描行的行号； 参数设置 4，返回定位轴轴号； 参数设置 5，返回扫描轴轴号

函数名	DLL_PulseTrigger		
函数原型	int DLL_PulseTrigger(unsigned int Address, int KG)		
函数功能	设置到位触发脉冲功能		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	KG	有符号 32 位整型	KG 为 1 则开启到位脉冲触发功能， KG 为 0 则关闭到位脉冲触发功能
返回值		有符号 32 位整型	返回 1 则设置成功， 返回 0 则未收到校验， 返回-1 则设置失败。

相关串口通讯格式如下所示：

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 位整型	/
		扫描模式	/	子模式选择	控制字节		/
发送	0x20	2 (扫描模式)	0	‘I’	1	I 模式扫描并启动	WO
				‘D’	1	D 模式扫描并启动	
				‘P’	1	P 模式扫描并启动	
				‘N’	1	N 模式扫描并启动	
		0x80 (到位触发模式)		0	1	到位触发模式并启动	
		*		*	0	停止扫描	
接收	0x20	高位			低位	确认收到	/

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型	/
发送	0x22	高位			低位	设置扫描起始坐标点	RW
接收	0x22	高位			低位	确认扫描起始坐标点	/
	0x50	0	0	0	76	超过软限位最小坐标	
		0	0	0	77	超过软限位最大坐标	

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型	/
发送	0x26	高位			低位	设置扫描触发间距，单位 counts	RW
接收	0x26	高位			低位	确认扫描触发间距	/
	0x50	0	0	0	70	触发间距过大	
		0	0	0	71	触发间距过小	

备注：

（1） 需注意速度与触发间距之间关系，防止信号触发异常。

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型	/
发送	0x27	高位			低位	设置扫描触发脉冲数量	RW
接收	0x27	高位			低位	确认扫描触发脉冲数量	/

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型	/
发送	0x28	高位			低位	设置扫描触发脉冲脉宽 单位 us 输入范围 5～999	RW
接收	0x28	高位			低位	确认扫描触发脉冲脉宽	/
	0x50	0	0	0	72	触发脉冲脉宽过大	
		0	0	0	73	触发脉冲脉宽过小	

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型	/
发送	0x29	高位			低位	设置扫描触发脉冲掩盖数量 输入范围为 0 到 2000	RW
接收	0x29	高位			低位	确认扫描触发 脉冲掩盖数量	/
	0x50	0	0	0	74	掩盖数量过大	
		0	0	0	75	掩盖数量过小	

备注：

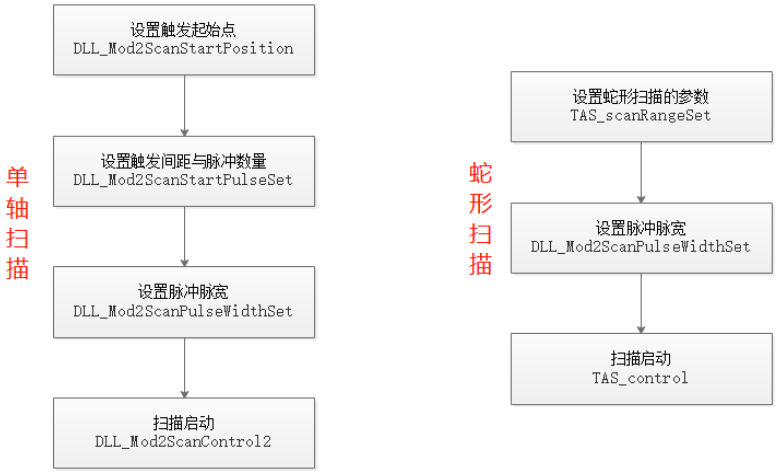
（1） 输入范围为 0 到 2000。

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0x9E	高位			低位	设置扫描速度 单位（×10）counts/ms 输入范围 2～100	RW
接收	0x9E	高位			低位	扫描速度	/
	0x50	0	0	0	33	设置的扫描速度过大	
		0	0	0	34	设置的扫描数量过小	

备注：

（1）输入范围为 2 到 100。

图 16.8 扫描 DLL 调用顺序为



十九、脉冲控制模式

19.1 脉冲控制说明

- 脉冲控制功能用于控制卡控制驱控器使用（类似步进电机控制器控制步进电机驱动器）。
- 分为两种连接方式，共阳接法跟差分接法。
- 需注意，驱控器默认配置为脉冲输出版本，需要进行电路硬件上（连接导通电阻）更改才能用于脉冲输入。

表 19.1 脉冲控制参数

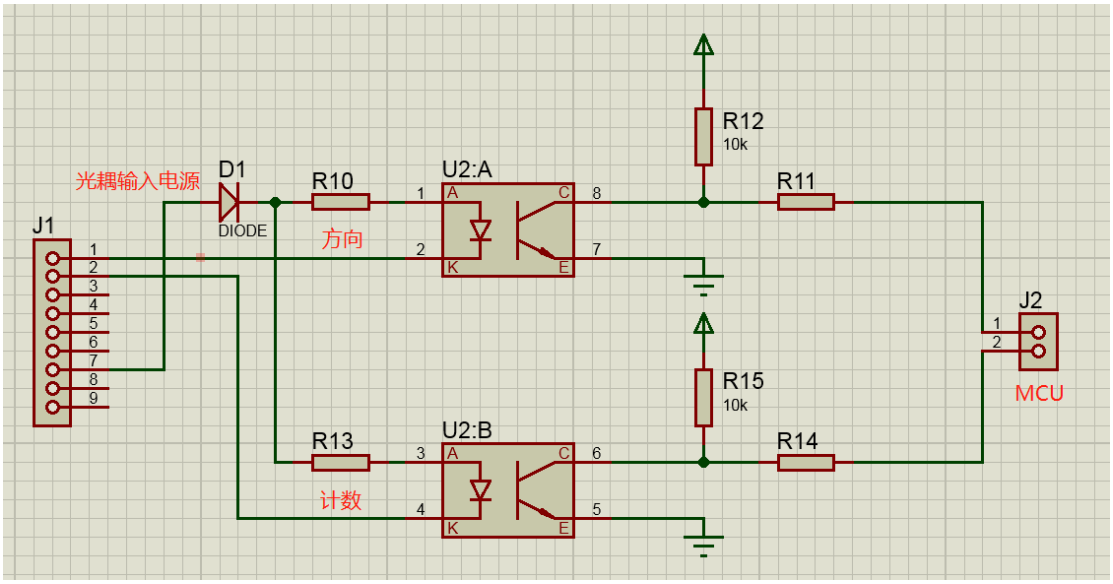
PCB 版本	输入电压	连接方式	光耦型号	最高频率 Mhz
V2.3		不支持		
V3.1	①直流 ②3.3V~30V， ③大于 5V 时，需外接限流电阻	共阳接法	H11L1	0.5
V3.5			EL0631	1
V3.1/V3.5+外接模块		差分接法		
JC-C03-P01				

备注：

（1）PCB 版本可通过查询 BootLoader 找到对应版本。

19.2 脉冲控制硬件原理

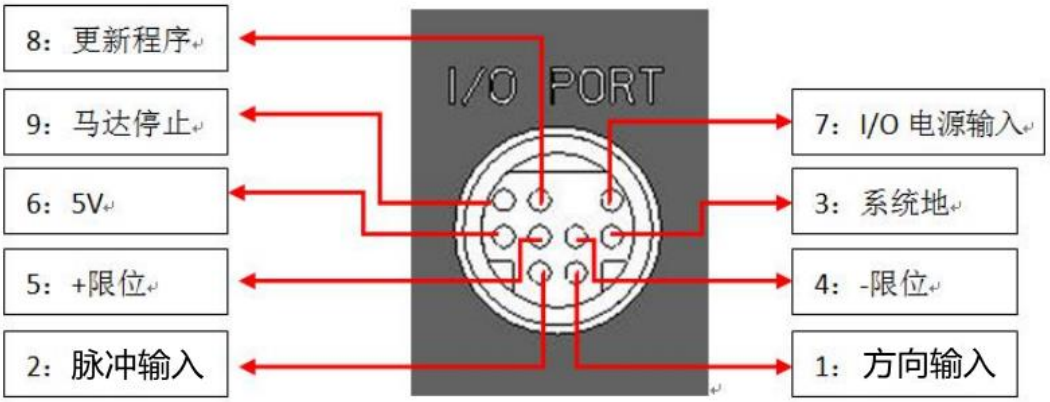
图 19.1 脉冲输入模型图



19.3 脉冲控制硬件连接

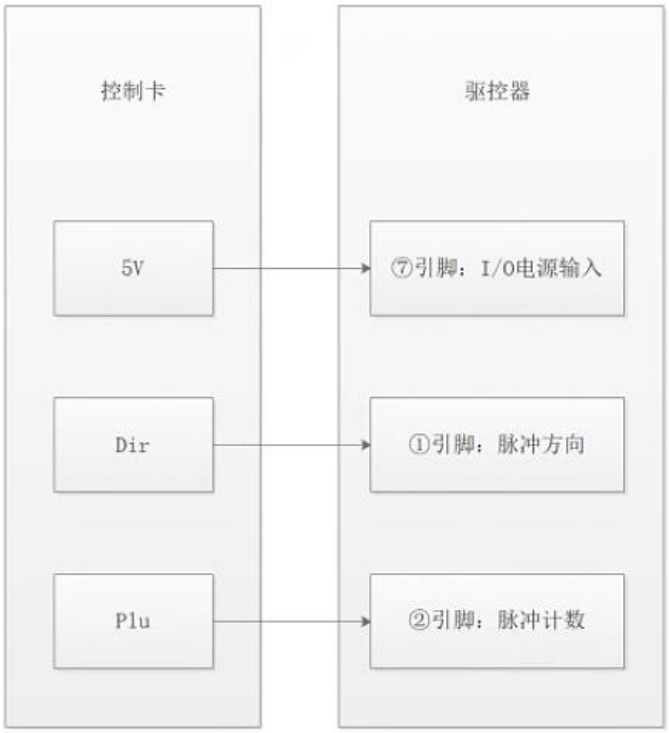
19.3.1 脉冲控制共阳接法

图 19.2 驱控器 I/O 接口定义图



使用到引脚号有：①②⑦。

图 19.3 驱控器与控制卡连接图（共阳接法）



19.3.2 脉冲控制差分接法

图 19.4 脉冲控制外接模块

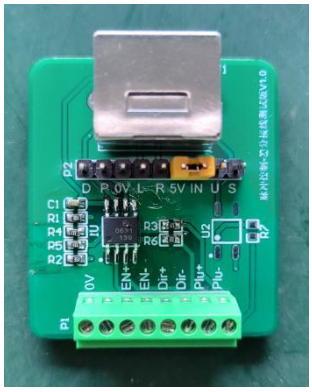
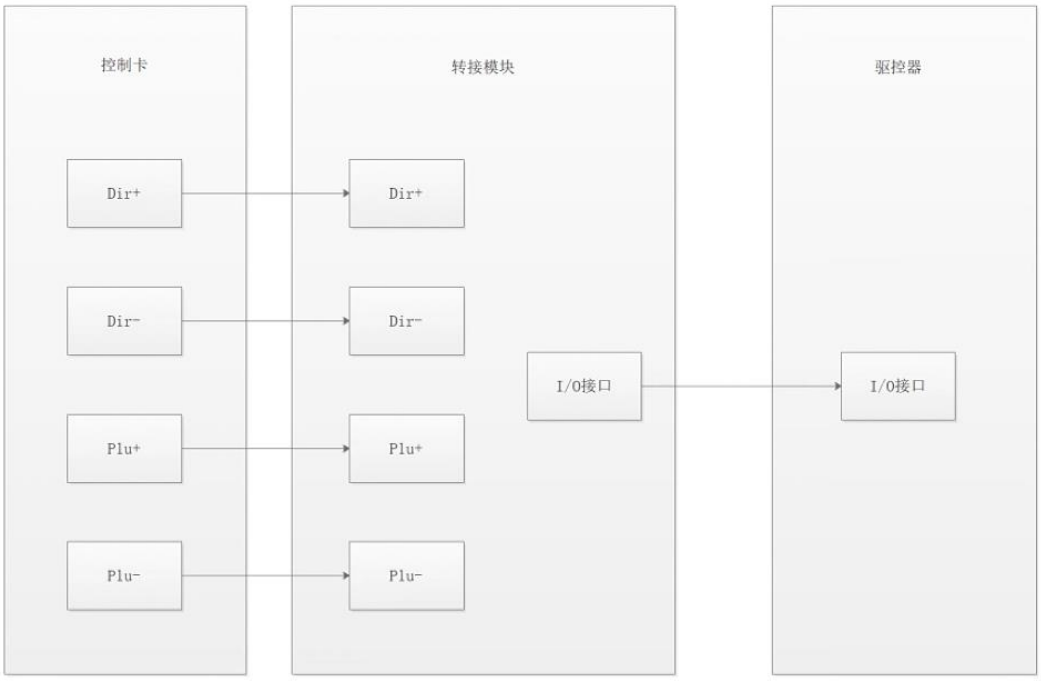


表 19.2 脉冲控制外接模块引脚定义表

符号	定义	说明
EN+	使能控制输入+	形成压差并导通，驱控器失能
EN-	使能控制输入-	
DIR+	方向输入+端	方向控制；形成压差并导通，标记驱动方向为正方向；不导通时，标记驱动方向为 - 方向
DIR-	方向输入-端	
PLU+	脉冲输入+	脉冲计数
PLU-	脉冲输入+	

图 19.5 驱控器与控制卡连接图（差分接法）



19.3.3 光耦 H11L1 参数

光耦型号为 H11L1（速率为 1Mhz），详细参数查看光耦说明书。
信号线上默认限流电阻阻值为 510Ω，注意需要根据光耦输入电源的电压，对信号线进行串接电阻。

在高频率使用下（空闲时为不导通），脉冲输入线路的限流电阻的关系：
①限流电阻过大，导致电流过小，光耦无法导通；
②限流电阻过小，导致电流过大，光耦经过导通后，脉冲为高电平时，光耦未完成关闭，持续导通。

从出现的现象进行分析：
①出现只计数 1 或者脉冲计数丢失，限流电阻过小
②未计数，限流电阻过大

19.3.4 光耦 H11L1 最大接收频率实际测试与分析

测试一参数：输入电源电压为 3.3V，空闲时，输入脉冲为高电平，光耦无导通，驱控器检测端电平为高（光耦逻辑反），发送脉冲总数为 320，逻辑电平 0V/3.3V。

表 17.3 H11L1 测试一结果

限流电阻 KΩ	输入脉冲频率 Mhz	接收到脉冲数	结果与分析
1.5	42/128（328Khz）	320	正常计数
	42/64（656Khz）	0	持续为高电平； 限流电阻过大，电流过小，光耦无法导通
1	42/128	320	正常计数
	42/64	12	起始部分正常，后续持续为低电平； 限流电阻偏小，电流过大，光耦无法关闭
1.05	42/64	320	正常计数
1.1	42/64	319	起始第 1 个脉冲丢失； 限流电阻偏大，导致起始时导通电流不够
1.22	42/64	318	起始第 1 个跟第 2 个脉冲丢失； 限流电阻偏大，导致起始时导通电流不够

测试二参数：输入电源电压为 3.3V，空闲时，输入脉冲为高电平，光耦无导通，驱控器检测端电平为高（光耦逻辑反），发送脉冲总数为 320，逻辑电平 0V/3.3V，输入脉冲频率为 42Mhz/64(656Khz)。

表 19.4 H11L1 测试二结果

限流电阻 KΩ	输入脉冲数	接收到脉冲数	结果与分析
1.05	320	320	正常计数
	1000	1000	正常计数
	10000	10000	正常计数

测试三参数：输入电源电压为 5V，脉冲引脚端接三极管 8550 输入端，三极管输出端接 0V，控制端接脉冲输出板的 I/O 引脚（使用三极管 8550 控制光耦导通）。空闲时，输入脉冲为高电平，光耦无导通，驱控器检测端电平为高（光耦逻辑反），发送脉冲总数为 10000，逻辑电平 0V/3.3V。

表 19.5 H11L1 测试三结果

限流电阻 KΩ	输入脉冲频率 Mhz	接收到脉冲数	结果与分析
1.5	42/256	10000	正常计数
	42/128	4	起始部分正常，后续持续为低电平； 限流电阻偏小，电流过大，光耦无法关闭
1.6	42/128	4	起始部分正常，后续持续为低电平； 限流电阻偏小，电流过大，光耦无法关闭
1.7～2.5	42/128	10000	正常计数
	42/64	1	限流电阻过小，电流过大，光耦无法关闭
2.8	42/128	10000	正常计数
	42/64	6	限流电阻过小，电流过大，光耦无法关闭
2.85	42/128	10000	正常计数
	42/64	9795~0	限流电阻过小，电流过大，光耦无法关闭
2.9～3	42/128	9999	起始第 1 个脉冲丢失； 限流电阻偏大，导致起始时导通电流不够
	42/64	0	限流电阻过大，电流过小，光耦无法打开

测试四参数：输入电源电压为 5V，脉冲引脚端接三极管 8050 输入端，三极管输出端接 0V，控制端接脉冲输出板的 I/O 引脚（使用三极管 8550 控制光耦导通）。空闲时，输入脉冲为低电平，光耦无导通，驱控器检测端电平为高（光耦逻辑反），发送脉冲总数为 10000，逻辑电平 0V/3.3V。

表 19.6 H11L1 测试四结果

限流电阻 KΩ	输入脉冲频率 Mhz	接收到脉冲数	结果与分析
2.5～4	42/256	10000	正常计数
	42/128	0	限流电阻过小，电流过大，光耦无法关闭
4.5	42/256	10000	正常计数
	42/128	1	限流电阻过小，电流过大，光耦无法关闭
4.6～4.65	42/256	9996	
	42/128	1	限流电阻过小，电流过大，光耦无法关闭
5	42/256	0	限流电阻过大，电流过小，光耦无法打开
	42/128	0	限流电阻过大，电流过小，光耦无法打开

测试五参数：输入电源电压为 5V。空闲时，输入脉冲为高电平，电流不足已导通光耦，驱控器检测端电平为高（光耦逻辑反），发送脉冲总数为 10000，逻辑电平 0V/3.3V。

表 19.7 H11L1 测试五结果

限流电阻 KΩ	输入脉冲频率 Mhz	接收到脉冲数	结果与分析
1～1.3	42/256	10000	正常计数
	42/128	1	限流电阻过小，电流过大，光耦无法关闭
1.4～1.5	42/128	2	限流电阻过小，电流过大，光耦无法关闭
1.8～2	42/128	10000	正常计数
	42/64	2	限流电阻过小，电流过大，光耦无法关闭
2.51	42/128	10000	正常计数
	42/64	9999	起始第 1 个脉冲丢失； 限流电阻偏大，导致起始时导通电流不够

图 19.6 正常计数

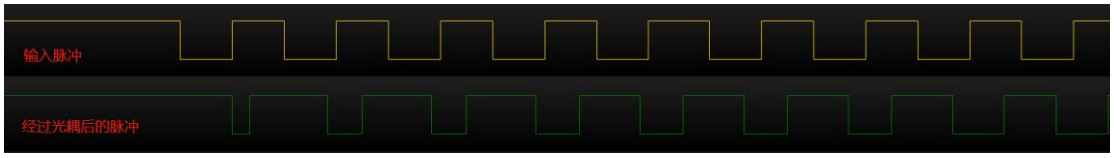


图 19.7 频率过高，限流电阻过大，0 个脉冲

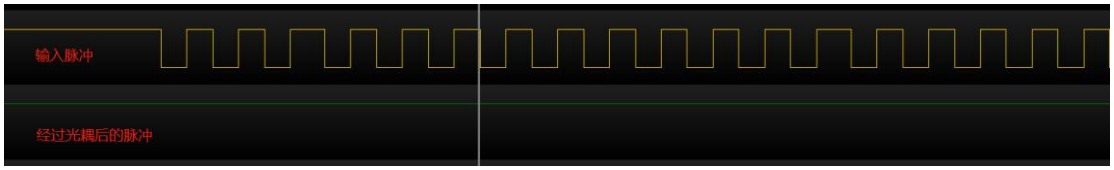


图 19.8 频率过高，限流电阻偏小，X 个脉冲

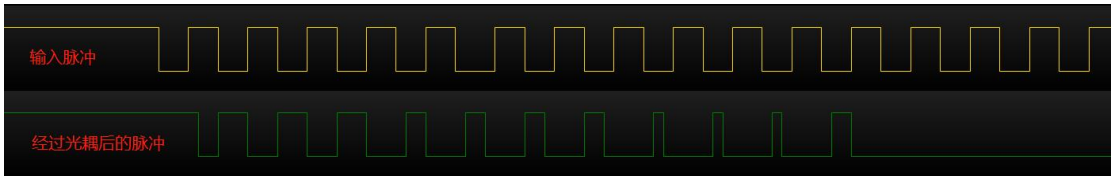
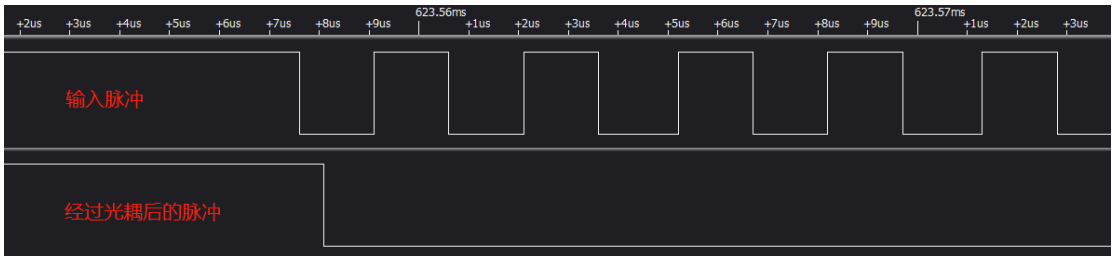


图 19.9 频率过高，限流电阻偏小，1 个脉冲



19.3.5 光耦 EL0631 参数

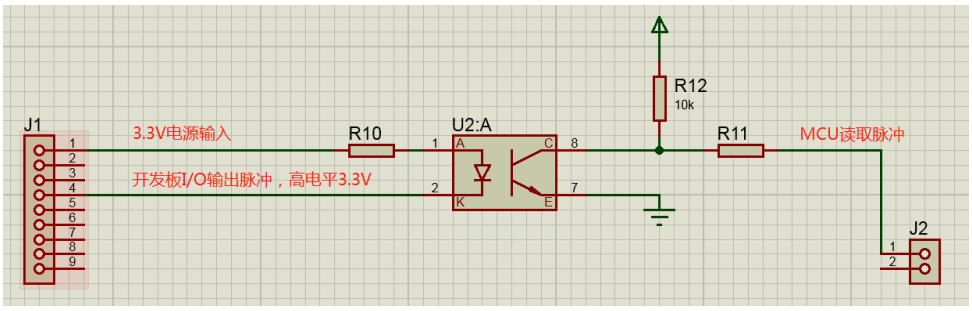
光耦型号为 EL0631，详细参数查看光耦说明书。
信号线上默认限流电阻阻值为 330Ω，注意需要根据光耦输入电源的电压，对信号线进行串接电阻。

在高频率使用下（空闲时为不导通），脉冲输入线路的限流电阻的关系：
①限流电阻过大，导致电流过小，光耦无法导通；
②限流电阻过小，导致电流过大，光耦经过导通后，脉冲为高电平时，光耦未完成关闭，持续导通。

从出现的现象进行分析：
①出现只计数 1 或者脉冲计数丢失，限流电阻过小
②未计数，限流电阻过大

19.3.6 光耦 EL0631 最大接收频率实际测试与分析

图 19.8 测试一原理图

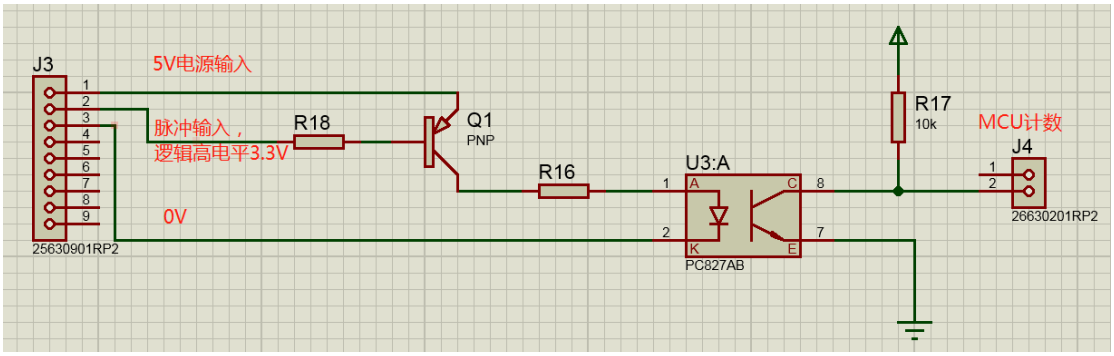


测试一参数：输入电源电压为 3.3V，空闲时，输入脉冲为高电平，光耦无导通，驱控器检测端电平为高（光耦逻辑反），发送脉冲总数为 10000，逻辑电平 0V/3.3V。

表 19.8 EL0631 测试一结果

限流电阻 Ω	输入脉冲频率 Mhz	接收到脉冲数	结果与分析
330	42/32	10000	正常计数
	42/16	10000	正常计数
	42/8	10000	正常计数
	42/4	7926	限流电阻稍偏小，电流稍大，光耦出现一定不关闭现象（超过光耦频率）

图 19.9 测试二原理图



测试二参数：输入电源电压为 5V，脉冲引脚端接三极管 8550 输入端，三极管输出端接 0V，控制端接脉冲输出板的 I/O 引脚（使用三极管 8550 控制光耦导通）。空闲时，输入脉冲为高电平，光耦无导通，驱控器检测端电平为高（光耦逻辑反），发送脉冲总数为 10000，逻辑电平 0V/3.3V。

表 19.9 EL0631 测试二结果

限流电阻 Ω	输入脉冲频率 Mhz	接收到脉冲数	结果与分析
330	42/32	10000	正常计数
	42/16	5259	限流电阻稍偏大，电流小，光耦出现一定不导通现象

备注：需要更高的输入频率，需要更改输入限流电阻

图 19.10 正常计数

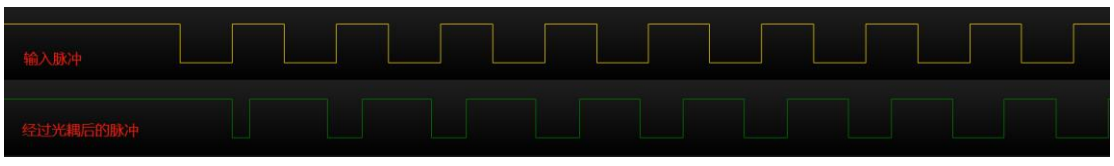


图 19.11 频率过高，限流电阻过大，0 个脉冲

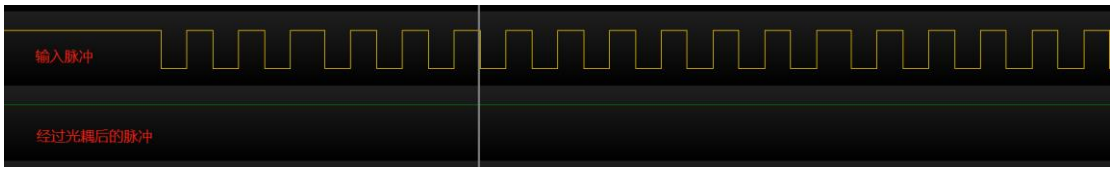


图 19.12 频率过高，限流电阻偏小，X 个脉冲

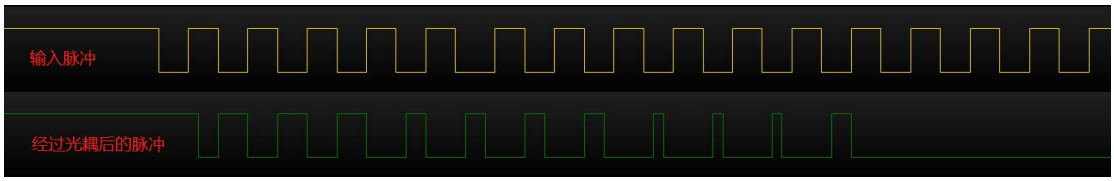


图 19.13 频率过高，限流电阻小，1 个脉冲

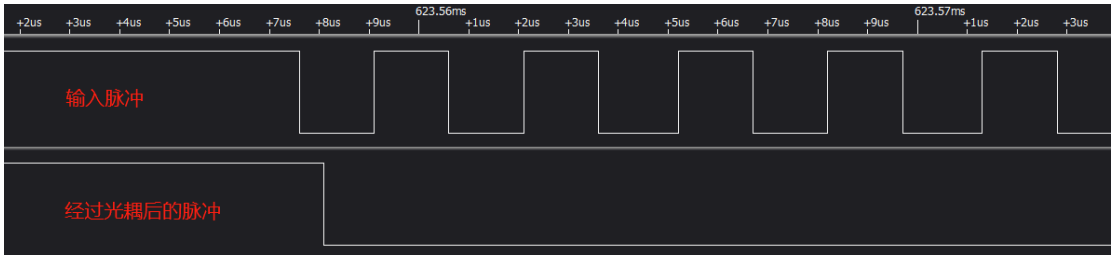
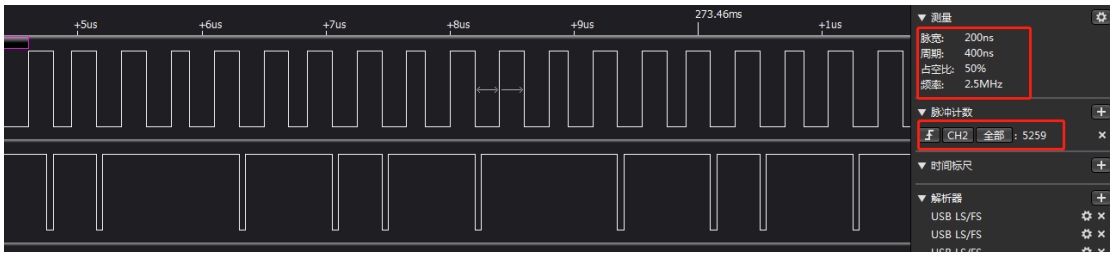


图 19.14 频率过高，限流电阻偏小，中间丢失脉冲，X 个脉冲



二十、开环使用

20.1 驱动方式说明

因为通信接口硬件上为 485 电路，通信过程需消耗时间，不能精确控制马达驱动脉宽，进而不能精确控制马达驱动位移量。为了解决这种现象，驱动器加入细调、中档、粗调三种使用方式。细调、中档用于预设控制，预设脉宽。

例如：上位机发送一条指令，大约需要 1ms（波特率 115200,10 个字节）。下位机（驱动器）判断接收结束的机制为判断不再接收到信息的间隔大于 1ms。如果上位机接收的机制一样，则一个通信周期至少 3ms。另外，通信过程中还可能会出现丢包现象（通信间隔越短，丢包概率越大，推荐通信间隔 50ms，不建议小于 20ms）。还有驱动马达位移大约 50nm，一般驱动脉宽小于 1ms。所以，使用通信无法保证精准控制驱动脉宽。

20.2 脉冲式

20.2.1 细调

- 使用脉冲方式进行驱动，通过设置脉冲频率、脉宽、幅值和脉冲数量控制位移量。
- 使用这种方式，驱动器将记录所驱动的脉冲数（正负代表方向），类似于步进电机。
- 将脉冲数作为位置参考量。
- 驱动器程序上，通过配置 MCU 的定时器，使用 PWM 方式控制脉冲频率与脉宽（消除程序运行引起的滞后，提高精度）。

相关 DLL 函数如下所示：

函数名	DLL_FineOpenDrive		
函数原型	int DLL_FineOpenDrive(unsigned int Address,int Num)		
函数功能	开环驱动的细调档位控制，用于精细控制驱动		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Num	有符号 32 位整型	脉冲数
返回值		有符号 32 位整型	返回 1 指令发送成功， 返回 0 未收到校验， 返回-1 指令发送失败。

函数名	DLL_SetFineOpenFre		
函数原型	int DLL_SetFineOpenFre (unsigned int Address,unsigned int Fre)		
函数功能	设置开环驱动的细调档位的频率		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Fre	无符号 32 位整型	频率（单位 hz）， 输入范围 1～1Khz。
返回值		有符号 32 位整型	返回 1 设置成功， 返回 0 未收到校验， 返回-1 设置失败。

函数名	DLL_SetFineOpenWidth		
函数原型	int DLL_SetFineOpenWidth (unsigned int Address, unsigned int Width)		
函数功能	设置开环驱动的细调档位的脉宽		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Width	无符号 32 位整型	脉宽（单位 us）， 输入范围 150～1000。
返回值		有符号 32 位整型	返回 1 设置成功， 返回 0 未收到校验， 返回-1 设置失败。

函数名	DLL_SetFineOpenAmp		
函数原型	int DLL_SetFineOpenAmp (unsigned int Address,unsigned int Amp)		
函数功能	设置开环驱动的细调档位的幅值		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Amp	无符号 32 位整型	幅值，输入范围 1～350。
返回值		有符号 32 位整型	返回 1 设置成功， 返回 0 未收到校验， 返回-1 设置失败。

相关串口通讯格式如下所示

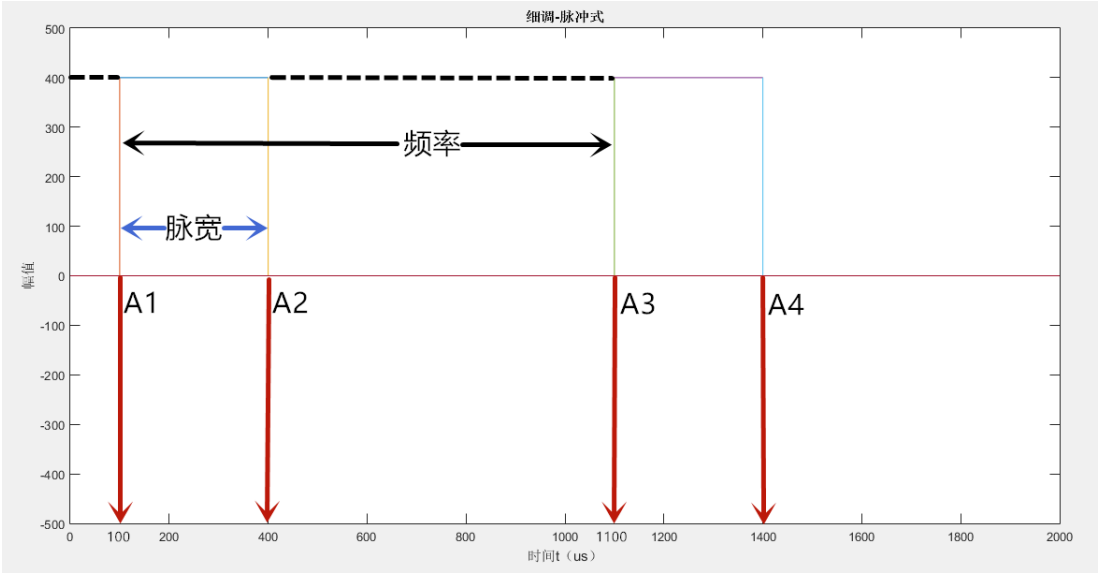
含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型	/
发送	0x61	高位			低位	开环细调档位频率 单位 0.01hz 输入范围 50～20000 （0.5～2Khz）	RW
接收	0x61	高位			低位	确认开环细调档位频率	/
	0x50	0	0	0	60	开环细调档位频率过大	
		0	0	0	61	开环细调档位频率过小	

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型	/
发送	0x62	高位			低位	开环细调档位脉宽 单位 10us 输入范围 1～50 （100～200us）	RW
接收	0x62	高位			低位	确认开环细调档位脉宽	/
	0x50	0	0	0	62	开环细调档位脉宽过大	
		0	0	0	63	开环细调档位脉宽过小	

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型	/
发送	0x63	高位			低位	开环细调档位幅值 输入范围 1～350	RW
接收	0x63	高位			低位	确认开环细调档位幅值	/
	0x50	0	0	0	64	开环细调档位幅值过大	
		0	0	0	65	开环细调档位幅值过小	

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	有符号 32 为整型	/
发送	0x68	高位			低位	开环细调档位驱动 ±代表方向 数值代表驱动的脉冲数量	WO
接收	0x68	高位			低位	确认开环细调档位驱动	/

图 20.1 开环细调驱动时序图



例 1（图 20.1）：

在 A1 时刻前，驱动器收到 3 条指令。第 1 条指令数据类型 0x61，数据区内容为 100000，则细调档位的脉冲频率为 1Khz（100000/100=1K）；第 2 条指令数据类型 0x62，数据区内容为 300，则细调档位的脉宽为 300us；第 3 条指令数据类型 0x63，数据区内容为 400，则细调档位的脉冲幅值为 400。（不设置，则沿用上一次的参数）

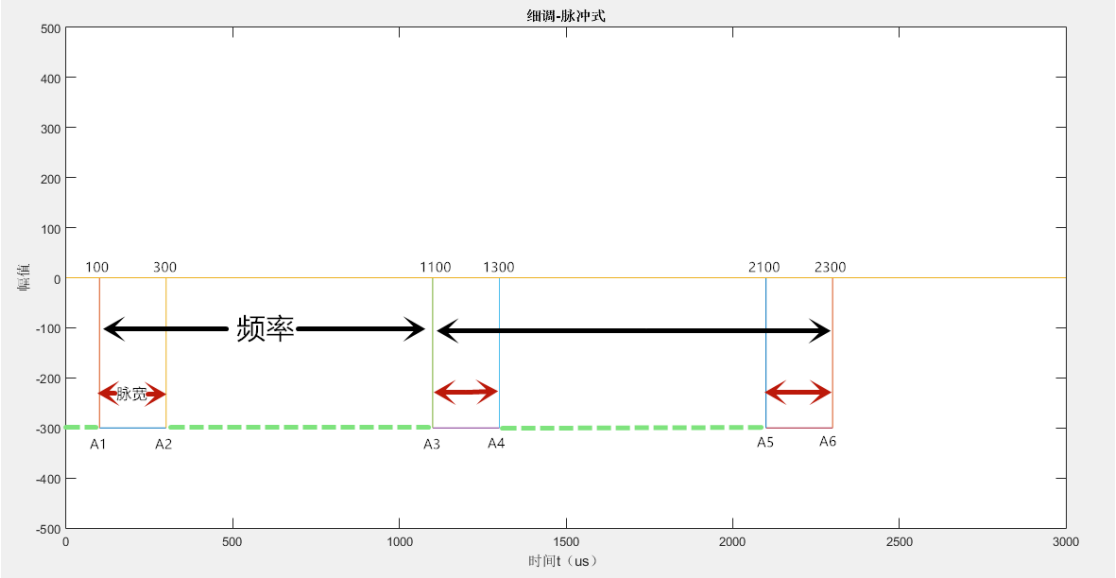
在 A1 时刻，驱动器收到指令（数据类型 0x83，数据区内容为 2），则驱动马达向“+”移动，输出脉冲数为 2，频率为 1Khz（1000us），脉宽为 300us，输出力度值为 400；

在 A2 时刻，停止驱动马达，输出脉冲数未达到预设值，等待下个周期；

在 A3 时刻，重新输出脉冲；

在 A4 时刻，停止驱动马达，输出脉冲数到达预设值，不再输出脉冲；

图 20.2 开环细调驱动时序图



例 2（图 20.2）：

在 A1 时刻前，驱动器收到 3 条指令。第 1 条指令数据类型 0x61，数据区内容为 100000，则细调档位的脉冲频率为 1Khz（100000/100=1K）；第 2 条指令数据类型 0x62，数据区内容为 200，则细调档位的脉宽为 200us；第 3 条指令数据类型 0x63，数据区内容为 300，则细调档位的脉冲幅值为 300。（不设置，则沿用上一次的参数）

在 A1 时刻，驱动器收到指令（数据类型 0x83，数据区内容为-3），则驱动马达向“-”移动，输出脉冲数为 3，频率为 1Khz（1000us），脉宽为 200us，输出力度值为 300；

在 A2 时刻，停止驱动马达，输出脉冲数未达到预设值，等待下个周期；

在 A3 时刻，重新输出脉冲；

在 A4 时刻，停止驱动马达，输出脉冲数未达到预设值，等待下个周期；

在 A5 时刻，重新输出脉冲；

在 A6 时刻，停止驱动马达，输出脉冲数到达预设值，不再输出脉冲；

20.2.2 粗调

设置的脉宽为 ms 级别，用于快速移动。

函数名	CoarseOpenDrive		
函数原型	int DLL_CoarseOpenDrive (unsigned int Address,unsigned int Width,int Amp)		
函数功能	开环驱动的粗调控制，用于控制快档驱动和中档驱动		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	Width	无符号 32 位整型	脉冲宽度，单位为 ms
	Amp	有符号 32 位整型	脉冲幅值，±代表驱动方向
返回值		有符号 32 位整型	返回 1 则指令发送成功， 返回 0 则未收到校验， 返回-1 则指令发送失败

表 20.2 粗调档位相关串口指令

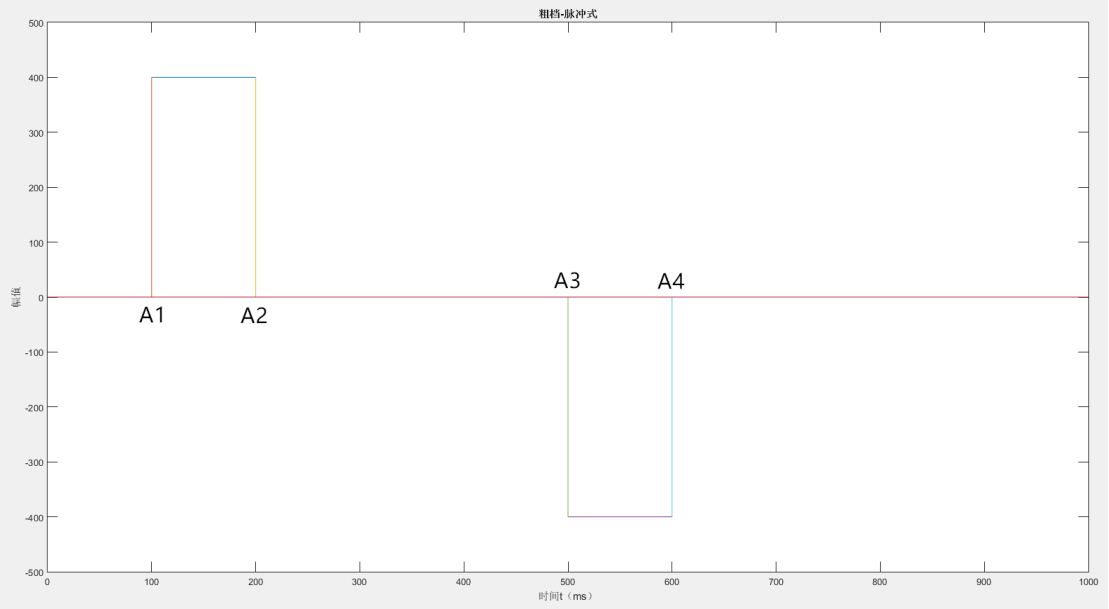
数据类型	数据区内容				数据区含义	读写属性
Byte3	Byte4	Byte5	Byte6	Byte7		
0xA8	脉冲宽度		脉冲幅值		粗调驱动	WO
	高位	低位	高位	低位		
	输入范围 0～65535		输入范围 -400～400			

使用数据类型为 0xA8 指令，用于粗调，输出脉冲个数为 1 个，并且不进行脉冲计数。

Byte4 和 Byte5 合并成一个无符号 16 位半字，用于设置脉冲宽度，单位为 ms；当该值为 0 时（脉冲幅值不等于 0 时），则不会主动停止驱动，直至设置脉冲幅值为 0。

Byte6 和 Byte7 合并成一个有符号 16 位半字，用于设置脉冲幅值，输入范围为-400 到 400，正负表示驱动方向；

图 20.3 开环粗调档位驱动时序图



例 3：

在 A1 时刻，驱动器收到指令（数据类型 0xA8；数据区内容，Byte4=0x00，Byte5=0x64，则不预设停止；Byte6=0x01，Byte7=0x90，则幅值为 400），则驱动马达向“+”移动，输出力度值为 400（当输出力度>0，值↑，输出力↑；当输出力度<0，值↓，输出力↑）。

在 A2 时刻，驱动器收到指令（数据类型 0xA8，数据区内容为 0），则停止驱动马达。

在 A3 时刻，驱动器收到指令（数据类型 0xA8；数据区内容，Byte4=0x00，Byte5=0x64，则不预设停止；Byte6=0xFE，Byte7=0x70，则幅值为-400），则驱动马达向“-”移动，力度值为 400。

在 A4 时刻，驱动器收到指令（数据类型 0xA8，数据区内容为 0），则停止驱动马达。

例 4：

在 A1 时刻，驱动器收到指令（数据类型 0xA8；数据区内容，Byte4=0x00，Byte5=0x00，则预设 100ms 停止；Byte6=0x01，Byte7=0x90，则幅值为 400），则驱动马达向“+”移动，输出力度值为 400（当输出力度>0，值↑，输出力↑；当输出力度<0，值↓，输出力↑）。

在 A2 时刻，驱动器计时结束，自动停止驱动马达。

在 A3 时刻，驱动器收到指令（数据类型 0xA8；数据区内容，Byte4=0x00，Byte5=0x00，则预设 100ms 停止；Byte6=0xFE，Byte7=0x70，则幅值为-400），则驱动马达向“-”移动，力度值为 400。

在 A4 时刻，驱动器计时结束，自动停止驱动马达。

20.3 正弦波

未开放使用

二十一、程序更新

21.1 外观区分不同硬件版本

图 21.1 驱控器 V2.3 版本顶视图

图 21.2 驱控器 V3.1 版本顶视图

图 21.3　驱控器 V3.5 版本顶视图

区别这 3 大版本的不同，可查看图片左下角区域，对应的芯片外观不一样。

21.2 更新说明

- 更新程序方式分为硬性更新和软性更新。
- 使用软性更新失败之后，可通过硬性更新。

表 21.1 为软件版本条件对应支持更新方式

更新方式	条件	
	BootLoader 版本	APP 版本
支持硬性更新	V1.0,V4.0 与 V4.0 以后	V3.4.2 以后
支持软性更新	,V4.0 与 V4.0 以后	V3.6.7 以后

每个版本 BootLoader 对应不同版本的 PCB 和可下载的 bin 文件大小。

表 21.2 不同版本对应可下载文件大小

BootLoader 版本	可下载的 bin 文件大小 byte	PCB 版本	更新事项
V1.0	40960	V2.3	
V3.0	61,440	V2.3	① 修改可接收文件大小。
V4.0	（60×1024）	V3.1	
V5.0	61,440	V2.3	① 增加分包更新程序模式。

V6.0	（60×1024）	V3.1	
V7.0	65000	V2.3	① 解决超过接收大小时，会自动覆盖旧程序，并且通讯收发冲突问题。 ② 修改可接收文件大小。
V8.0		V3.1	
V11.0	81920	V2.3	①修改可接收文件大小。
V12.0		V3.1	
V13.0		V3.3	
V14.0	81920	V2.3	①修复分包更新功能的一些问题
V15.0		V3.1	
V16.0		V3.3	
V17	81920	JC-C03-P01	

备注：

(1) 其他未标明的咨询工程师；

(2) V14 版本之前的版本可能存在不支持分包分送（有些驱控器程序上配置引起）；

(3) V11 到 V13 版本存在异常，不能下载大小÷10K 为单数的程序（bin 文件），比如 59K÷10K=5.9，取整为 5，5 为单数，运行异常。

注意事项：

（1）注意程序文件要对应上相应的驱控器硬件版本。

比如 V2.3 版本驱控器的程序文件名为 APP_V2.3_xxx.bin，V2.3 表示硬件版本，xxx 为软件版本；

V3.1 版本驱控器的程序文件名为 APP_V3.1_xxx.bin，V3.1 表示硬件版本，xxx 为软件版本；

V3.3 版本驱控器的程序文件名为 APP_V3.3_xxx.bin，V3.3 表示硬件版本，xxx 为软件版本。

（2）注意程序文件是否超过可接收的大小。查看表格表 3.2 查询可接收的大小。

21.3 BootLoader 流程图

图 21.4 BootLoader_V1.0 流程图

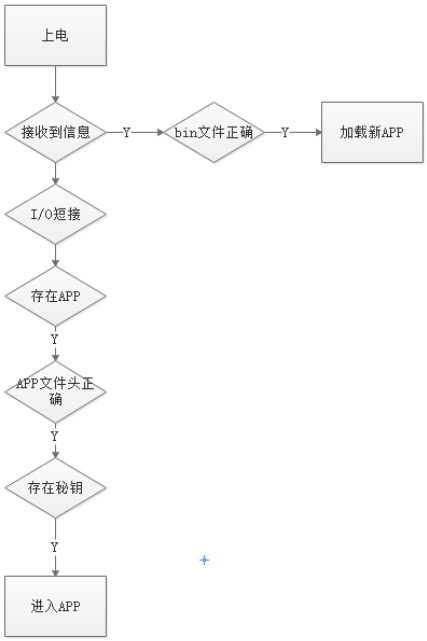


图 21.5 BootLoader_V4.0 流程图

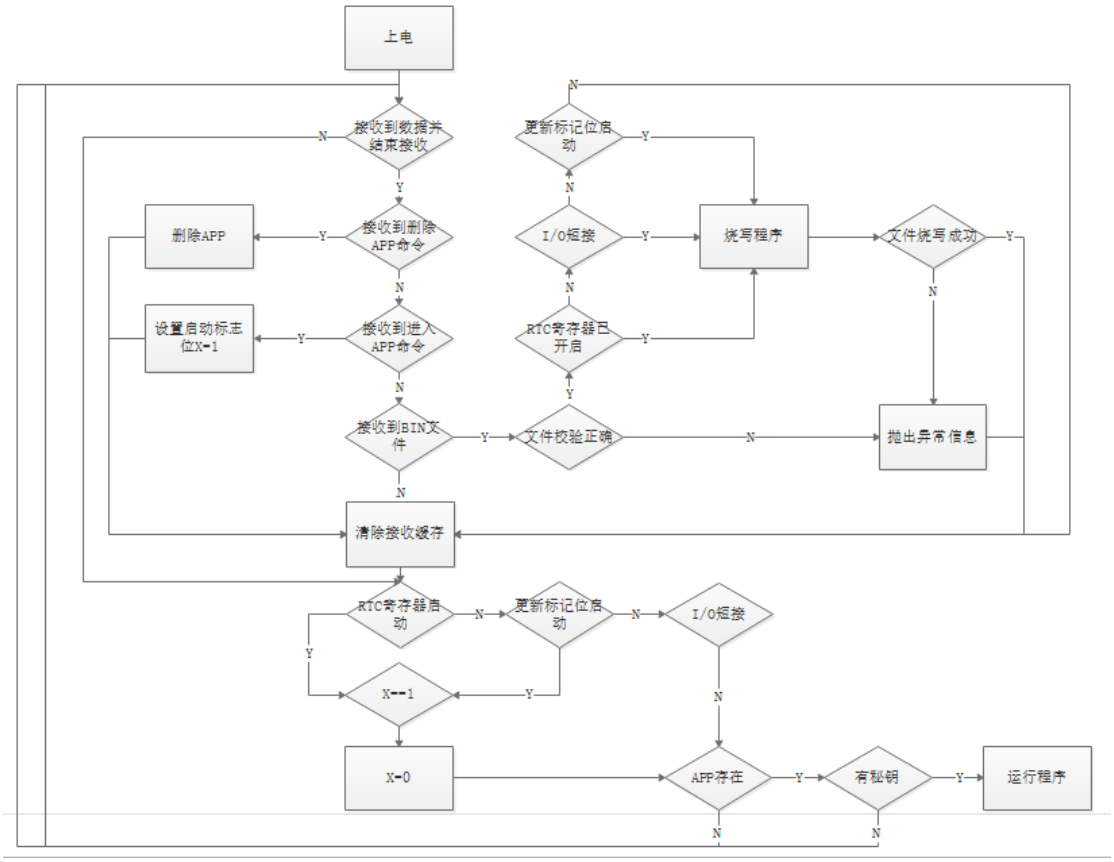


图 21.6 BootLoader_V6.0 流程图

21.4 识别 BootLoader 版本

进入 APP 程序后，使用查询 BootLoader 版号指令查询。

21.5 BootLoader 模式下驱控器显示灯状态

表 21.3 BootLoader 模式下显示灯对应状态表

	更新 APP 模式	无秘钥状态	无 APP 状态
Alarm1	0.5s 闪烁一次	0.1s 闪烁一次	1s 闪烁一次
Alarm2	熄灭		

21.6 更新流程说明

21.6.1 更新说明

在支持条件下，建议使用《JC-B1 更新程序软件》下的“更新模式二”方式（分包发送模式）进行更新，防止出现异常情况。

21.6.2 硬件更新方式

简易流程为：
（1） 断电
（2） 连接特定 I/O 引脚
（3） 上电（红灯闪烁）
（4） 打开串口上位机，配置，并连接
（5） 打开并发送 bin 文件
（6） 校验接收的文件大小，校验正常执行下一步，否则重新发送 bin 文件
（7） 拔出 I/O 引脚连接

具体使用流程如下所示。

准备工作：
① 用带有发送 bin 文件功能的串口软件和程序文件（APP.bin）；
② 串口软件配置设置（选择对应 COM 口，波特率 115200，其他如图 21.7 所示）；

图 21.7 串口助手配置界面

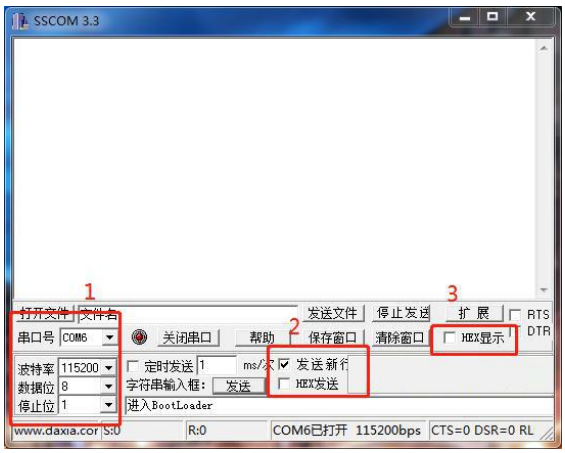
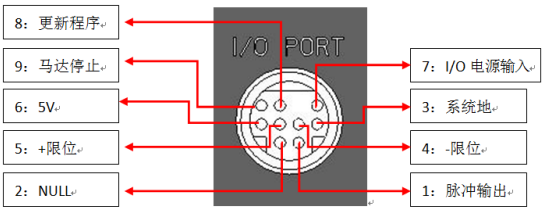


图 21.8 驱控器 I/O 接口图



③ 将需要更新程序的驱控器的 PS2 插口中 3 和 8 引脚短接，6 跟 7 引脚短接，重新上电后（或者发送重启指令），将进入 BootLoader 模式时

备注：

1、 每次只能更新一个轴，所以只能同时只将一个驱控器进行短接操作（可以都上电）；

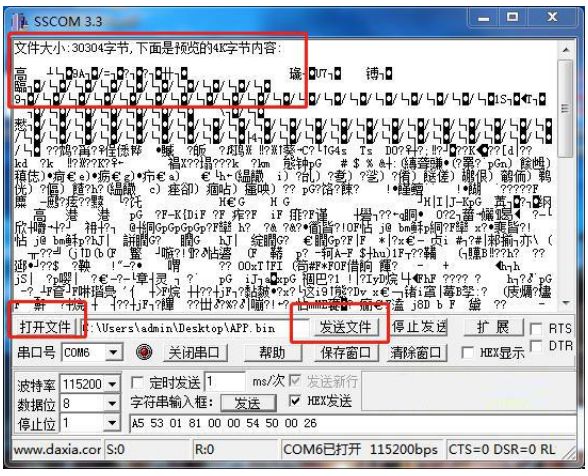
2、 重启等待过程中，红灯常亮；

3、 进入 BootLoader 模式后，红色闪烁。

更新程序步骤：

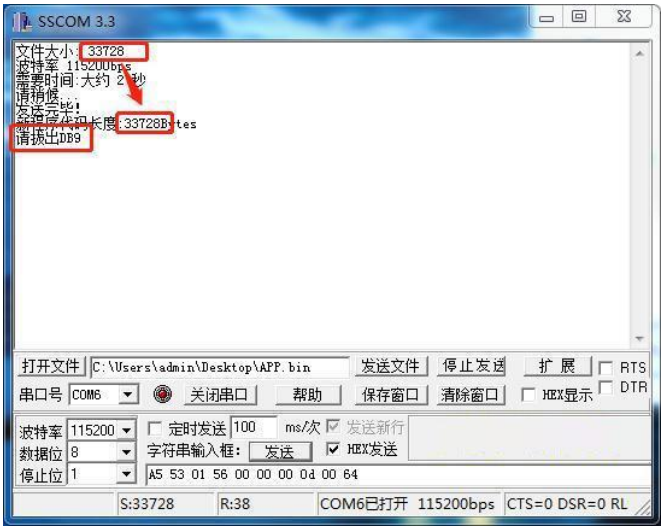
① 选择发送的文件（默认名字 APP.bin）；在“打开文件”下，选择发送的文件（不要发送），选择完之后会出现如图 21.9 情况，之后按下“清除窗口”。

图 21.9 串口助手发送程序文件



② 按下“发送文件”，上位机发送 bin 文件。
③ 接收到“\r\n 用户程序接收完成! 代码长度:%dBytes\r\n”后，校验文件大小与反馈回的“新程序代码长度”大小是否一致（图 21.10），如果相同则下载成功，否则重新发送一次文件。

图 21.10 串口助手发送文件后界面



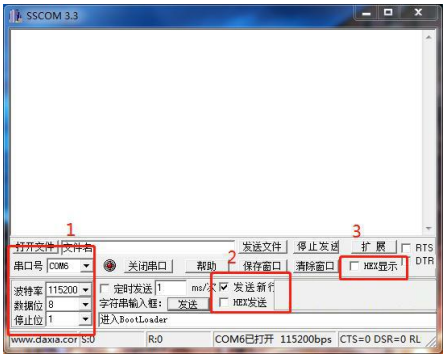
④ 取消 PS2 口短接，则进入新程序（DB9 接入时，绿色闪烁，否则红灯常亮），可以通过查询软件版本指令查询软件版本确认程序更新完成。

21.6.3 软件更新方式一

简易流程为：
（1）上电。
（2）打开串口上位机。
（3）发送指令。指令格式为“轴”+“X” +“更新程序”，X 为轴号，例如“轴 1 更新程序”；
（4）等待红灯闪烁再执行下一步；
（5）发送 bin 文件；
（6）校验接收的文件大小，校验正常执行下一步，否则重新发送 bin 文件；
（7）发送指令“运行 APP”。

具体使用流程如下所示。
准备工作：
① 用带有发送 bin 文件功能的串口软件和程序文件（APP.bin）
② 串口软件配置设置（选择对应 COM 口，波特率 115200，其他如图 21.11 所示）

图 21.11 串口助手配置界面



③ 驱控器已上电，正常进入运行模式（红灯常亮或者绿灯闪烁）

备注：

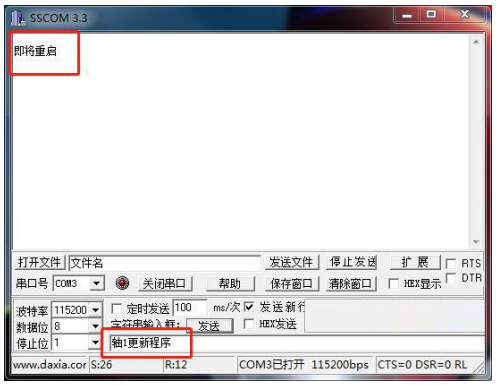
1、 重启等待过程中，红灯常亮；

2、 进入 BootLoader 模式后，红色闪烁。

更新程序步骤：

① 发送指令，指令格式为“轴”+“X” +“更新程序”，X 为轴号，例如，将编号为轴 1 的驱控器进行程序更新，如图 21.12 所示，在方框内输入“轴 1 更新程序”；随后反馈窗口显示“即将重启”（此时驱控器红灯常亮，进入 BootLoader 模式后，红灯闪烁）。

图 21.12 输入指令



② 选择发送的文件（默认名字 APP.bin）；在“打开文件”下，选择发送的文件（不要发送），选择完之后会出现如图 21.13 情况，之后按下“清除窗口”。

图 21.13 发送程序文件

③ 按下“发送文件”，上位机发送 bin 文件。

图 21.14 接收完成时显示界面

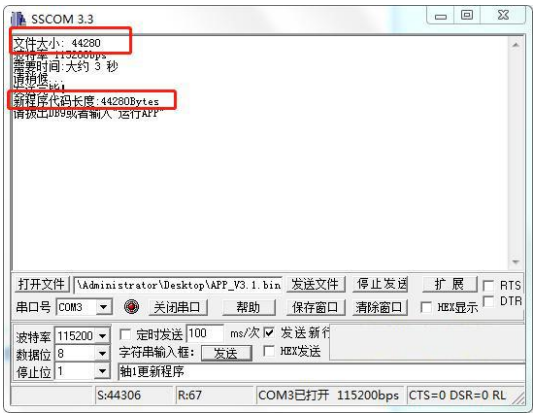
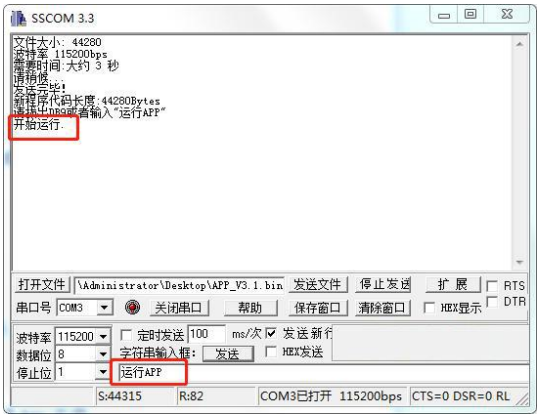


图 21.15 执行程序后显示界面



④ 接收到“\r\n 用户程序接收完成! 代码长度:%dBytes\r\n”后，校验文件大小与反馈的“新程序代码长度”大小是否一致（图 21.14），如果相同则下载成功，否则重新发送一次文件。

⑤ 发送指令“运行 APP”（图 21.15）（进入运行模式后，红灯常亮或者绿色闪烁）。

21.6.4 软件更新方式二（分包更新）

准备工作：

① 使用特定的更新程序软件 JC-B1 和程序文件（APP_Vx.x_Vxxx.bin）；

② 更新程序软件设置（选择对应 COM 口并打开，如图 21.17 所示）

③ 驱控器已上电，正常进入运行模式（红灯常亮或者绿灯闪烁）

备注：

1、 重启等待过程中，红灯常亮

2、 进入 BootLoader 模式后，红色闪烁

软件说明：
软件所在文件夹包含，上位机本体（.exe 文件）、动态链接库（.dll 文件）和要更新的驱控程序文件（.bin 文件），详情可见图 21.16。

图 21.16 更新程序软件包

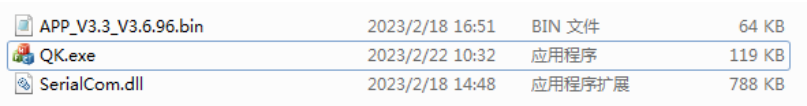


图 21.17 更新程序软件连接界面

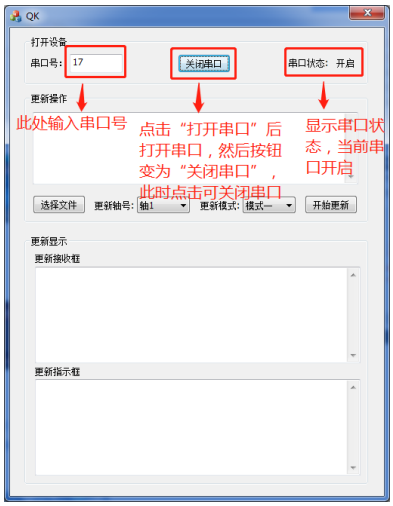


图 21.18 更新程序软件配置完成界面

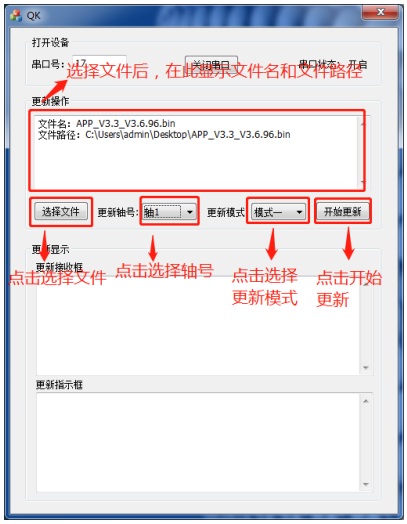


图 21.19 更新程序软件文件配对完成界面

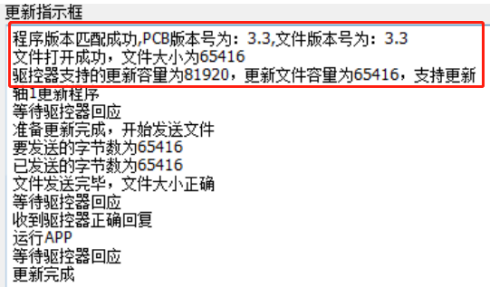


图 21.20 更新程序软件更新过程信息显示

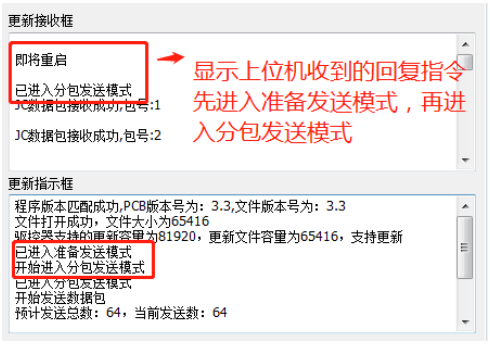


图 21.21 驱控器接收完成信息显示

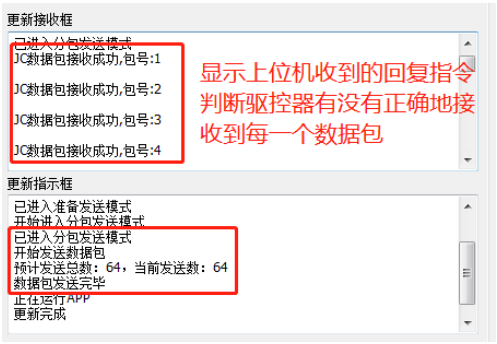
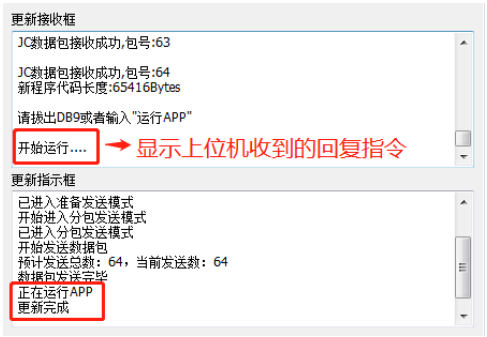


图 21.22 更新完成信息显示



更新程序步骤：

① 点击“选择文件”，选择要发送的程序文件，加载完成后，显示相应文件信息，如图 21.18。如果文件版本与硬件版本和文件大小不对应，将出现不匹配提示。

② 选择“模式二”，点击“开始更新”，等待发送成功。

③ 成功更新。

备注：

1、不能修改程序文件，包括文件名，在更新前会校验当前驱控器是否支持该更新文件，修改文件后可能会导致无法再用该文件更新上位机。

2、更新模式二如果发送文件出现异常后，可以断开驱控器电源，驱控器将运行旧程序。

二十二、其他参数说明

22.1 查询驱控器流水码

相关 DLL 函数如下所示：

函数名	DLL_GetPipelineCode		
函数原型	int DLL_GetPipelineCode(unsigned int Address)		
函数功能	查询机器流水码		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
返回值		有符号 32 位整型	查询成功返回对应值， 查询失败返回-1。

相关串口通讯格式如下所示：

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0x40	高位			低位	驱控器流水码	RW
接收	0x40	高位			低位	驱控器流水码	/
	0x50	0	0	0	94	设置错误	

备注：

（1）查询需要将通讯指令中查询参数位置 1。

（2）“设置错误”为设置流水码需要权限。

22.2 查询驱控器版本（软件版本，硬件版本，BootLoader 版本）

相关 DLL 函数如下所示：

函数名	DLL_AskVersionlum		
函数原型	int DLL_AskVersionlum (unsigned int Address,unsigned char BRY)		
函数功能	查询机器流水码		
	变量名	变量类型	说明
形参	Address	无符号 32 位整型	驱控器的轴号（1,2,3...）
	BRY	无符号 8 位整型	输入“B”查询 BootLoader 版本号， 输入“R”查询软性版本号， 输入“Y”查询硬性版本号。
返回值		有符号 32 位整型	查询成功返回对应值， 无接收返回 0， 查询失败返回-1。

相关串口通讯格式如下所示：

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
		硬件版本		软件版本			/
发送	0xA1	0	0	0	0	查询软件/硬件版本	RO
接收	0xA1	高位	低位	高位	低位	软件/硬件版本	/

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0x5F	0	0	0	0	查询 BootLoader 版本	RO
接收	0x5F	高位			低位	BootLoader 版本	/

备注：查询需要将通讯指令中查询参数位置 1。

22.3 将当前位置置零

对应 DLL 函数为 DLL_SetZeroPosition;
相关串口通讯格式如下所示：

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0x8A	0	0	0	0	置零	WO
接收	0x8A	0	0	0	0	收到置零	/

22.4 闭环细调档位参数

对应 DLL 函数为《DLL_SetStepValue》，《DLL_GetStepValue》。

22.5 设置波特率

相关串口通讯格式如下所示：

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0x6D	高位			低位	波特率， 输入范围 2400～256000	RW
接收	0x6D	高位			低位	波特率	/
	0x50	0	0	0	153	设置过小	
		0	0	0	154	设置过大	

备注：需要存储（参数存储说明）后重启生效。

22.6 复位

相关串口通讯格式如下所示：

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0XA0	0x55	0xAA	0x55	0xAA	复位	WO
接收	0XA0	0x55	0xAA	0x55	0xAA	收到复位	/
	0x50	0	0	0	99	输入错误	

22.7 查询驱控器 MCU 温度

相关串口通讯格式如下所示：

含义	数据类型	数据区				数据区含义	读写属性
编号	Byte3	Byte4	Byte5	Byte6	Byte7	无符号 32 为整型	/
发送	0X4E	0	0	0	0	查询温度	RO
接收	0X4E	高位			低位	温度值 需要将读出值除 100	/

二十三、异常排查

- 排查异常方法主要两种：排除法和交叉比较法。
- 异常主要分为两种：通讯异常，定位异常。

23.1 通讯异常

- 通讯异常表现为，应用程序连接不上驱控器。

- 软性问题

（1）串口驱动未安装。可通过设备管理器查看是否为已接入，但识别不出设备（出现感叹号），如果是则需要安装驱动。

（2）串口已被占用。可能为其他应用程序已打开被占用，或者为上一次应用程序奔溃，导致串口未被释放。可重新插拔 USB，或者插入电脑另外的 USB 口。

- 硬性问题

（1） 未上电。

（2） 未插入 USB。

（3） 应用程序处理异常，导致收发冲突。需重启驱控器。

（4） 通讯线异常。换新的通讯线测试。

- 具体可查看“串口打开异常说明书”。其他原因咨询工程师。

23.2 定位异常

- 排除思路查看排除框架图。
- 当 alarm1 灯常亮红，为定位异常（alarm 灯的颜色与闪烁频率对应的状态查看“指示灯”）。
- 异常的可能原因：

（1） 位置反馈与马达接口没有对应接入（多轴使用下）。
轴 1（X）的马达与反馈接入轴 1（X）驱控器（查看标识）

（2） 位置反馈异常。
现象：与上位机连接正常情况下，失能，使位移台台面左右缓慢来还移动，通过上位机读取位置信息，如果位置信息没有随之增减，则暂定原因为位移台。如果还有另外的驱控器，可换个驱控器进行测试此位移台，现象一样，则基本判断为位置反馈异常。

（3） 马达异常。
现象：开环测试或者闭环测试时关闭检测，驱动时，电流偏小或偏大（不同马达不同，需咨询工程师），则需要进行下一步测试。马达为容性负载，测量马达的容值，所以需测量 DB9 的 3 号跟 4 号引脚的容值，4 号跟 5 号引脚的容值，具体容值咨询工程师。如果还有另外的位移台，可换个驱控器进行测试此位移台，现象一样，则可能为马达异常，需进行其他步骤排除。

（4） 驱控器参数异常。
现象：关闭异常检测进行驱动，如果能正常驱动，则可能是驱控器参数的问题，需咨询工程师进行调试。
（5） 驱控器硬件异常。
（6） 其他需咨询工程师进行调试

图 23.1 排查异常流程图